

PE Practical Electronics

The UK's premier electronics, computing and maker magazine

Circuit Surgery
Implementing DSP
with a microcontroller

Audio Out
Cascode JFETs and BJTs
in a direct injection box

Precision Electronics
Bringing analog and digital
together: ADCs & DACs

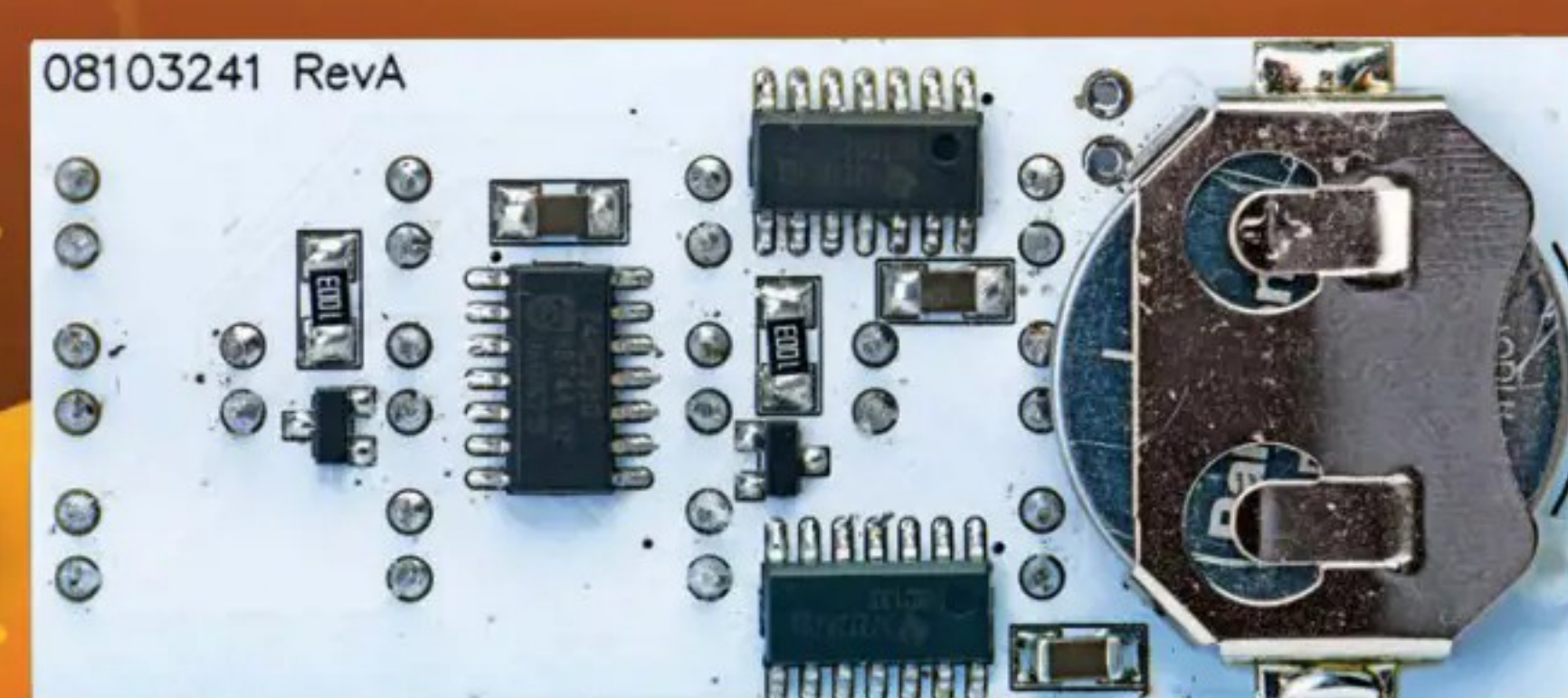


The Styloclone
A simple electronic
musical instrument

**Dual Hybrid
(switch-mode + linear)
Bench Supply**



**Two kinds of
Mini LED Dice**



PLUS!

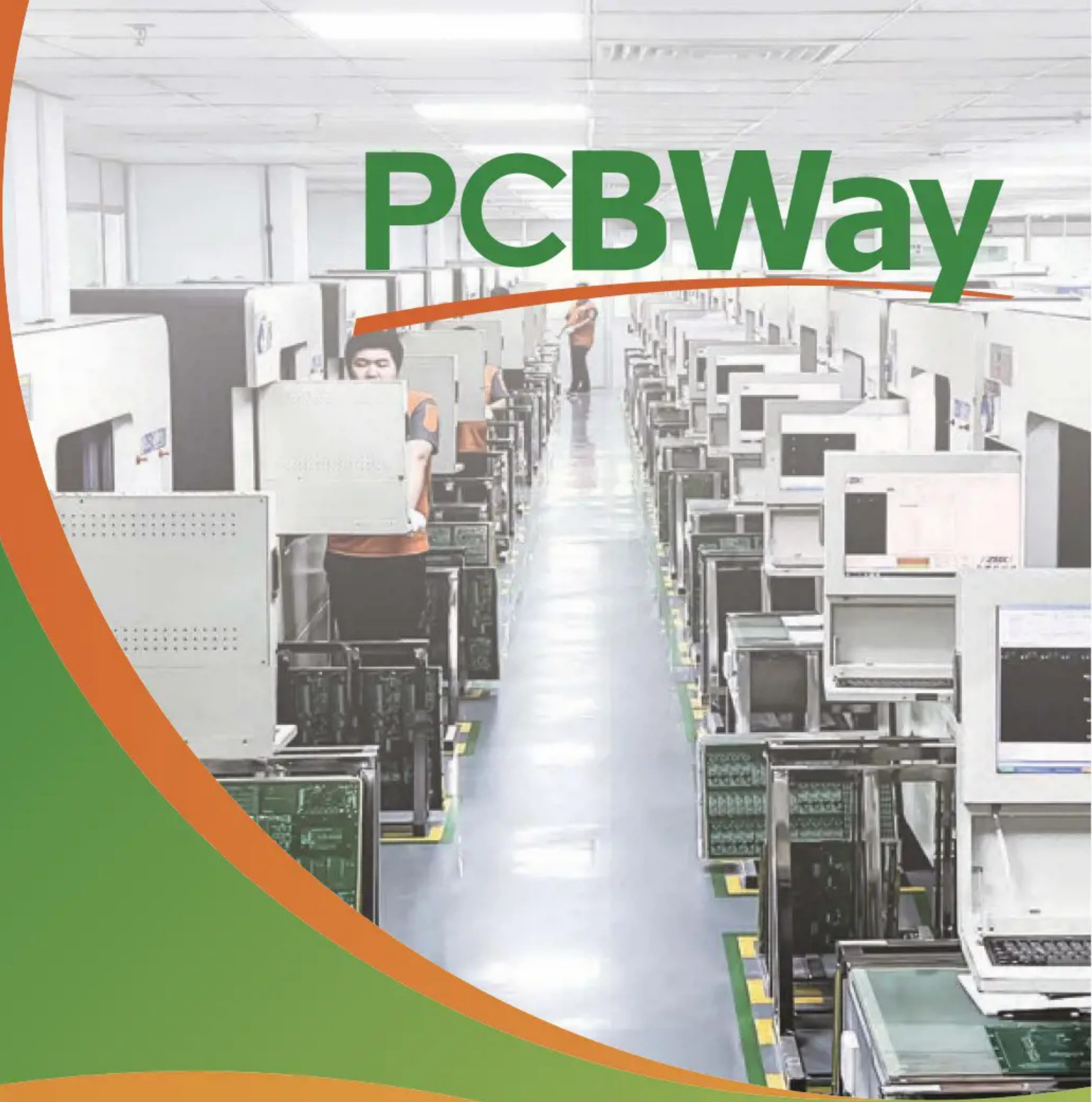
Techno Talk – Humanoid robots may be a reality soon
Cool Beans – shift registers and LED bargraphs
The History of Electronics – our final instalment

Jun 2025 £5.99

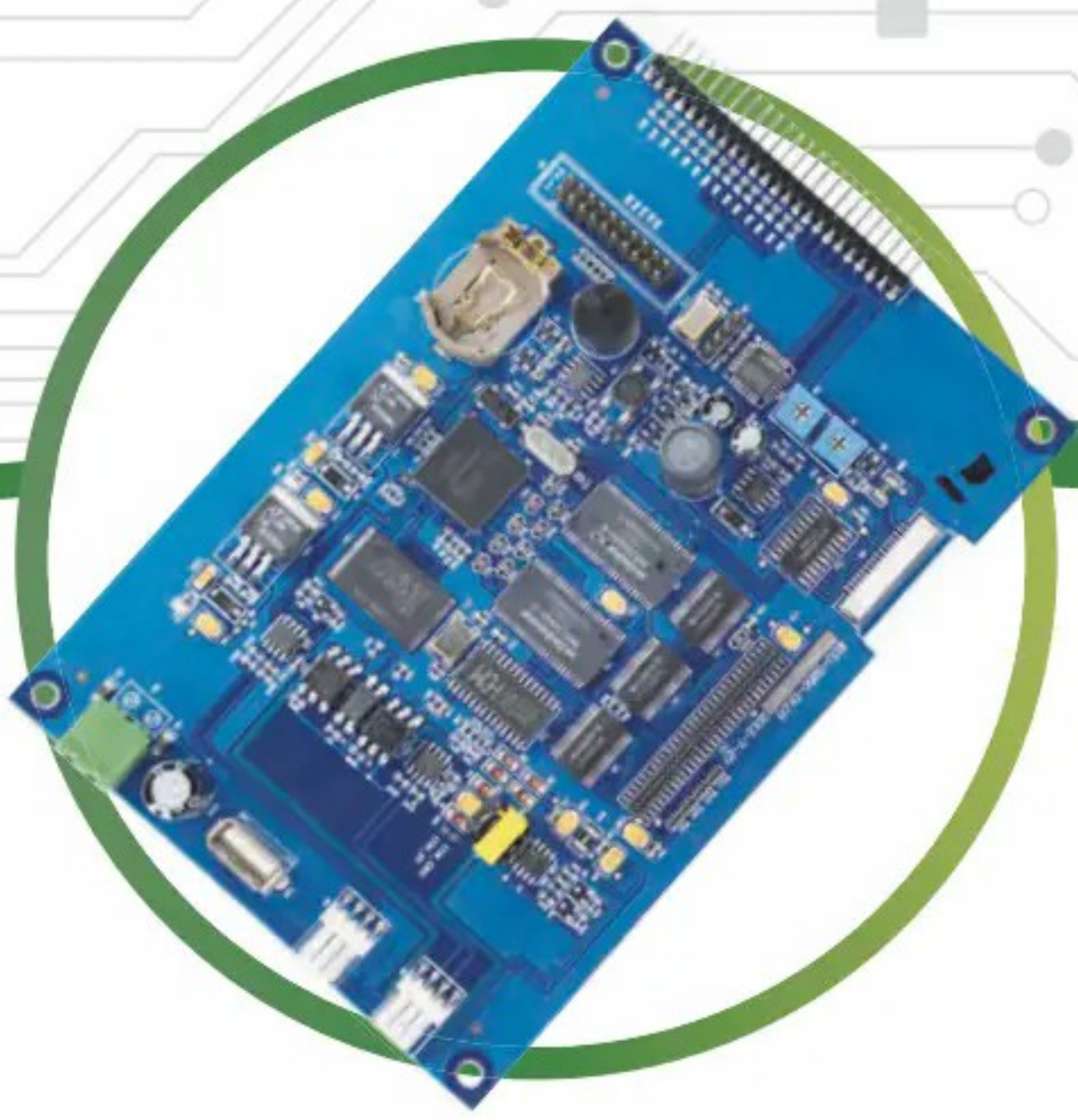


PCBWay

FULL-SERVICE ELECTRONICS MANUFACTURER

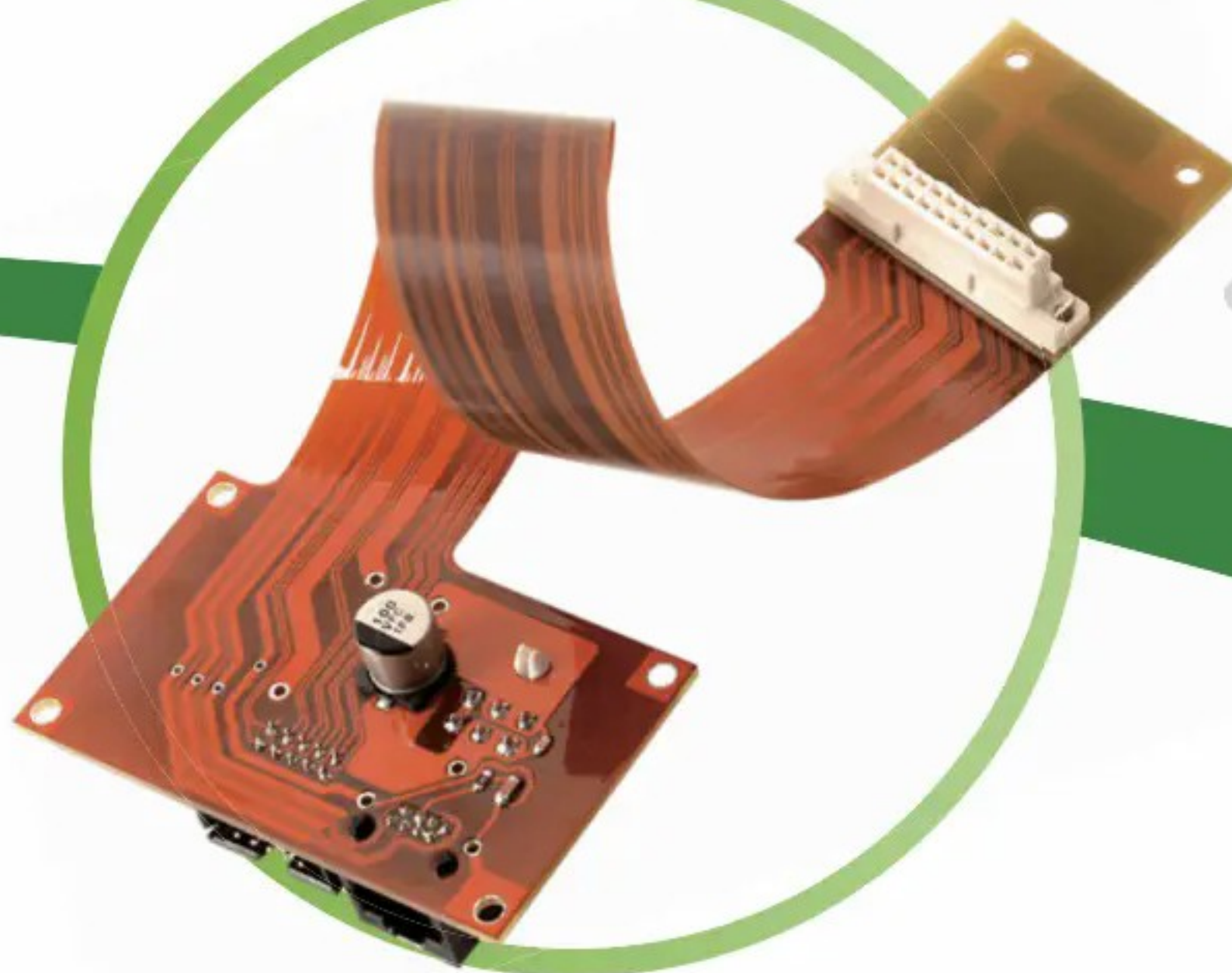


PCB Capabilities:



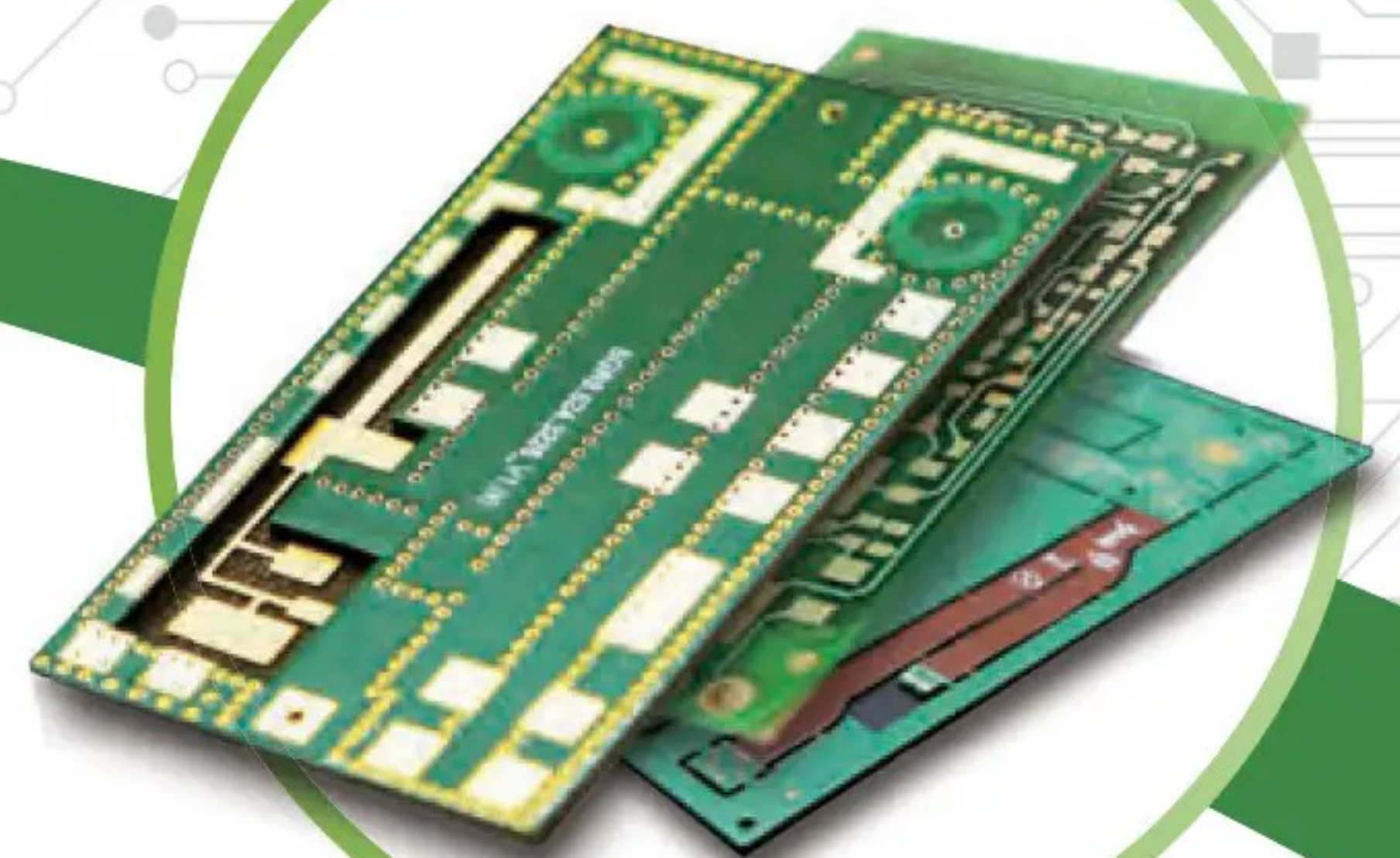
PCB Assembly From \$29

- Improved assembly precision
- Short turnaround for assembly
- Flexible support



Flex, Rigid-Flex PCB From \$46.74

- Up to 16 layers Flex circuits
- Polyimide Flex, PET, High Frequency
- 0.025-≥0.4 mm FPC thickness



**Special
price!**

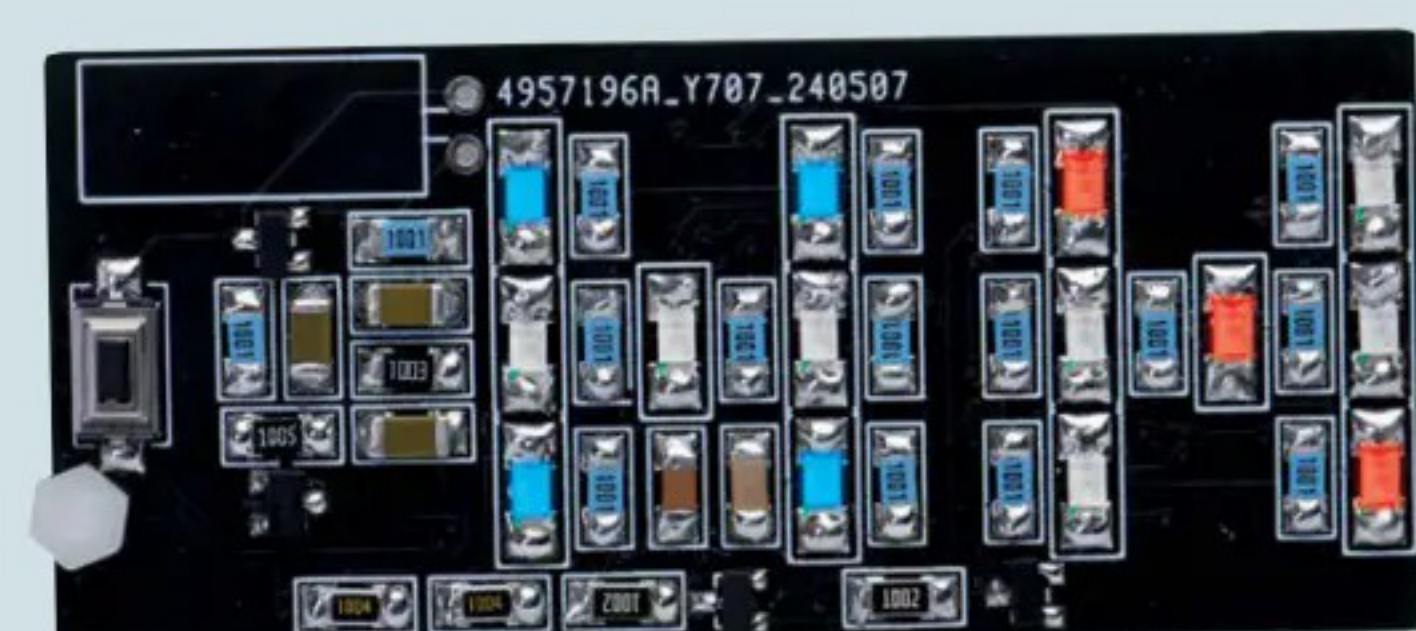
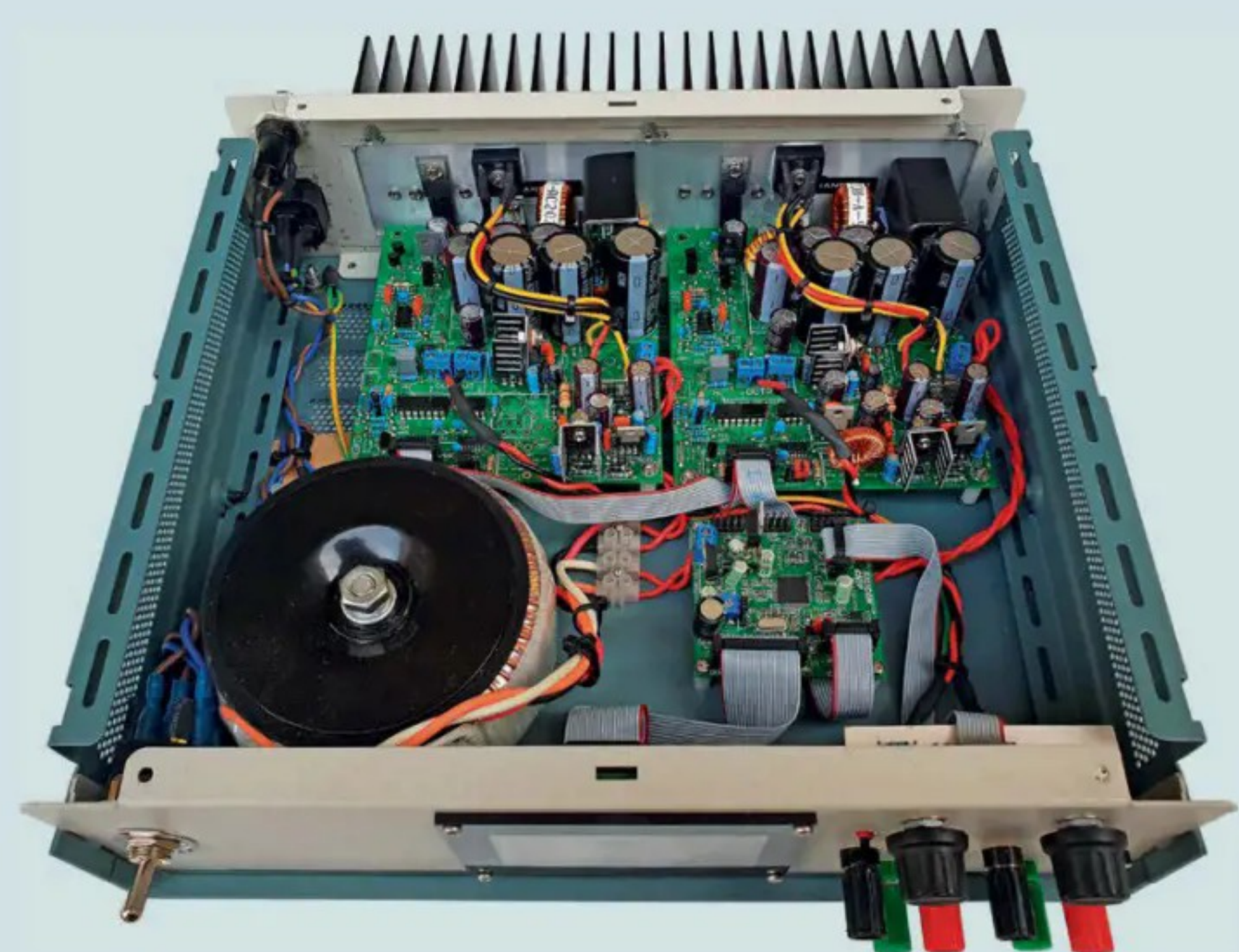
PCB Prototype From \$5/10 pcs

- Multiple color options
- 24-hour production time
- Worldwide delivery



 www.pcbway.com

 service@pcbway.com



Made in Australia & the UK.

Written in Britain, Australia,
the USA and Ireland.

Read everywhere.

© Silicon Chip Publications Pty Ltd 2025

Copyright in all drawings, photographs, articles, technical designs, software and intellectual property published in *Practical Electronics* is fully protected, and reproduction or imitation in whole or in part are expressly forbidden.

The July 2025 issue of *Practical Electronics* will be published on Thursday, 5th of June 2025 (see p80).

Projects and Circuits

The Styloclone musical instrument by Phil Prosser 3

This homage to a musical classic is an excellent project for beginners. It's also fun for the musically inclined. The whole project fits on one easy-to-build circuit board that can be mounted in a case or built as a free-standing project.

Intelligent Dual Hybrid Power Supply, part one by Phil Prosser 20

This power supply has two outputs, each capable of up to 25V DC at 5A. They can be connected in series and ganged up to form a dual tracking supply, and both outputs are controlled using a graphical LCD screen, two rotary encoders and two buttons.

Dual Mini LED Dice by Nicholas Vinen 42

This article is a blend of the old and the new. It was inspired by a design from 1994 but has been updated to use more modern parts (still with discrete logic) and runs from a 3V coin cell. As we have used mostly SMDs, it will easily fit in your pocket.

The Skill Tester 9000, part two by Phil Prosser 64

This retro game is a fun and educational project to create a dexterity-based game with nine difficulty levels, a health bar graph, a timer and four different sound effects. It is based mainly on 4000-series logic and all through-hole parts on a single circuit board.

Series, Features and Columns

The History of Electronics, part six by Dr David Maddison 13

Over the last five issues, we have described many individuals and organisations that made vital contributions to electronics. This sixth and final part in the series covers innovations from universities, companies and other organisations since 1962.

Circuit Surgery by Ian Bell 34

Topics in digital signal processing – Implementing DSP on a microcontroller, part one.

Audio Out by Jake Rothman 50

A phantom-powered direct injection (DI) box using a cascoded JFET/BJT pair.

Techno Talk by Max The Magnificent 54

Humanoid robots may not be the stuff of science fiction much longer.

Max's Cool Beans by Max The Magnificent 56

Weird & Wonderful Arduino Projects, part 6: shift registers and 8-segment displays.

Precision Electronics, part six by Andrew Levido 70

These days, most analog circuitry eventually interfaces with the digital realm, such as with a microcontroller, so we also need to understand the precision approach to analog-to-digital and digital-to-analog converters (ADCs & DACs).

Regulars and Services

Editorial 2

Don't tin wires going into terminal blocks

Reader services – Editorial and Advertising Departments 2

Subscribe to Practical Electronics and save money 33

Practical Electronics PCB Service 77

PCBs for Practical Electronics projects

Classified ads and Advertiser index 79

Next month! – highlights of our next issue of Practical Electronics 80

Renewals paid by PayPal, Visa and Mastercard are now handled via our website:
www.electronpublishing.com

Editorial offices

Electron Publishing (Aust.) Tel +61 2 9939 3295
Unit 1, 234 Harbord Road Email pe@pemag.au
Brookvale NSW 2100 Web www.electronpublishing.com
Australia

Address mail to:

Electron Publishing (Australia)
PO Box 194, Matraville NSW 2036
Australia

Advertising enquiries

+61 2 9939 3295 pe@pemag.au

Editor	Nicholas Vinen
Publisher	Nicholas Vinen
Digital subscriptions	Stewart Kearn Tel 07918 614662
Online Editor	Alan Winstanley
Web Systems	Kris Thain
Production	Bao Smith
Technical staff	Tim Blythman, John Clarke

Print subscriptions

Practical Electronics Subscriptions
PO Box 6337
Bournemouth BH1 9EH Tel 01202 087631
United Kingdom Email pesubs@selectps.com

Technical enquiries

We regret that technical enquiries cannot be answered over the telephone. We are unable to offer any advice on the use, purchase, repair or modification of commercial equipment or the incorporation or modification of designs published in the magazine.

Questions about articles or projects should be sent to the editor by email: pe@pemag.au

Projects and circuits

All reasonable precautions are taken to ensure that the advice and data given to readers is reliable. We cannot, however, guarantee it and we cannot accept legal responsibility for it.

Some projects and circuits published in *Practical Electronics* employ voltages that can be lethal. Do not build, test, modify or fix any mains-powered equipment unless you fully understand the safety aspects involved and you use an RCD (GFCI) adaptor.

Component supplies

Silicon Chip Publications may offer kits or other parts for making our projects, but not in all cases. When kits are not available, readers will need to find and source parts themselves. We advise readers to check that all parts are still available before commencing any project in a back-dated issue.

Advertisements

Although the proprietors and staff of *Practical Electronics* take reasonable precautions to protect the interests of readers by ensuring as far as practicable that advertisements are bona fide, the magazine and its publishers cannot give any undertakings in respect of statements or claims made by advertisers, whether these advertisements are printed as part of the magazine, or in inserts. The Publishers regret that under no circumstances will the magazine accept liability for non-receipt of goods ordered, or for late delivery, or for faults in manufacture.

Transmitters/bugs/telephone equipment

We advise readers that certain items of radio transmitting and telephone equipment which may be advertised in our pages cannot be legally used in the UK. Readers should check the law before buying any transmitting or telephone equipment, as a fine, confiscation of equipment and/or imprisonment can result from illegal use or ownership. The laws vary from country to country; readers should check local laws.

Don't tin wires going into terminal blocks

This seems to be a slightly controversial and misunderstood topic. There is a temptation to tin the end of a wire before you insert into a terminal block, especially a screw terminal. Doing so stops stray strands from sticking out and makes inserting the wire easier.

However, there are serious problems with doing so, and you really shouldn't. In fact, this practice is actively discouraged or even prohibited by most wiring standards, including BS 7671, AS/NZS 3000, and related IEC and UL standards.

The problems relate to mechanical stability and long-term reliability. Unlike copper, solder is a soft metal with a relatively low melting point. Over time, especially in the presence of heat and pressure, it can 'cold flow' – gradually deforming under the clamping force of a terminal screw. This can lead to a loose connection, increased contact resistance and, in extreme cases, overheating or arcing.

Also, when you insert a bare stranded wire into a terminal and tighten the screw, the strands compress and spread slightly. This creates a wide, gas-tight contact area, improving both electrical conductivity and mechanical grip. Tinning prevents that from happening. The solder locks the strands together, turning them into a pseudo-solid core that can't adapt to the clamping surface. The result is a reduced contact area, less secure grip and higher resistance.

Solder is also less conductive than copper, so the connection with the tinned wire not only has a lower contact surface area, but also higher inherent resistance. This makes for a potentially hot and unreliable joint.

If you're dealing with fine-stranded wire or worried about stray strands, the correct solution is not solder, it's a ferrule. These are metal sleeves, usually made of tin-plated copper, that crimp over the end of a stranded wire to hold it together. A properly crimped ferrule allows the terminal screw to compress the ferrule itself, maintaining a secure, stable connection over time. It also preserves the benefits of a gas-tight contact.

Ferrules are not expensive – decent kits with crimpers and assorted sizes are readily available. Once you try them, you'll probably never go back.

**Nicholas Vinen, Electron Publishing (Australia)*
Publisher & Editor,
Practical Electronics Magazine**

* a division of SILICON CHIP Publications Pty Ltd.

The Styloclone

Project by Phil Prosser



This reinvention and homage to a musical classic is an excellent project for starters. It's also lots of fun for the musically inclined, especially those interested in old-school instruments. The whole project fits on one easy-to-build circuit board that can be mounted in a case or built as a free-standing project!

This project is simple enough to be ideal for people learning electronics, but useful enough to be fun for all ages. A Stylophone is a very simple musical instrument that can play a single note at a time, driven by a stylus (or pen). A unique feature of a Stylophone is that it uses tracks on the PCB to form the instrument's keys.

The original Dubreq Stylophone was released in 1968. While it has never become as popular as, say, the electric guitar, it definitely made a mark on popular music! Notable uses of a Stylophone are at the start of David Bowie's "Space Oddity" and throughout the Tornados' instrumental, "Telstar". So, while simple, the distinctive sound of this instrument has a real place in music.

The name "Styloclone" indicates

it is not a real Stylophone; it is more an homage to the original instrument, drawing inspiration from it. Our version draws a lot from the 1970s design of the Stylophone, and keeps the essentials such as the PCB tracks forming the keyboard.

As an engineer, my immediate reaction to this project was to 'gold plate' it, allowing it to do things that no sensible person would want. However, that would miss the essence of the Stylophone.

We could have based it on a microcontroller, allowing all manner of clever stuff, including fancy waveforms and effects. Alternatively, we could 'keep the purity'.

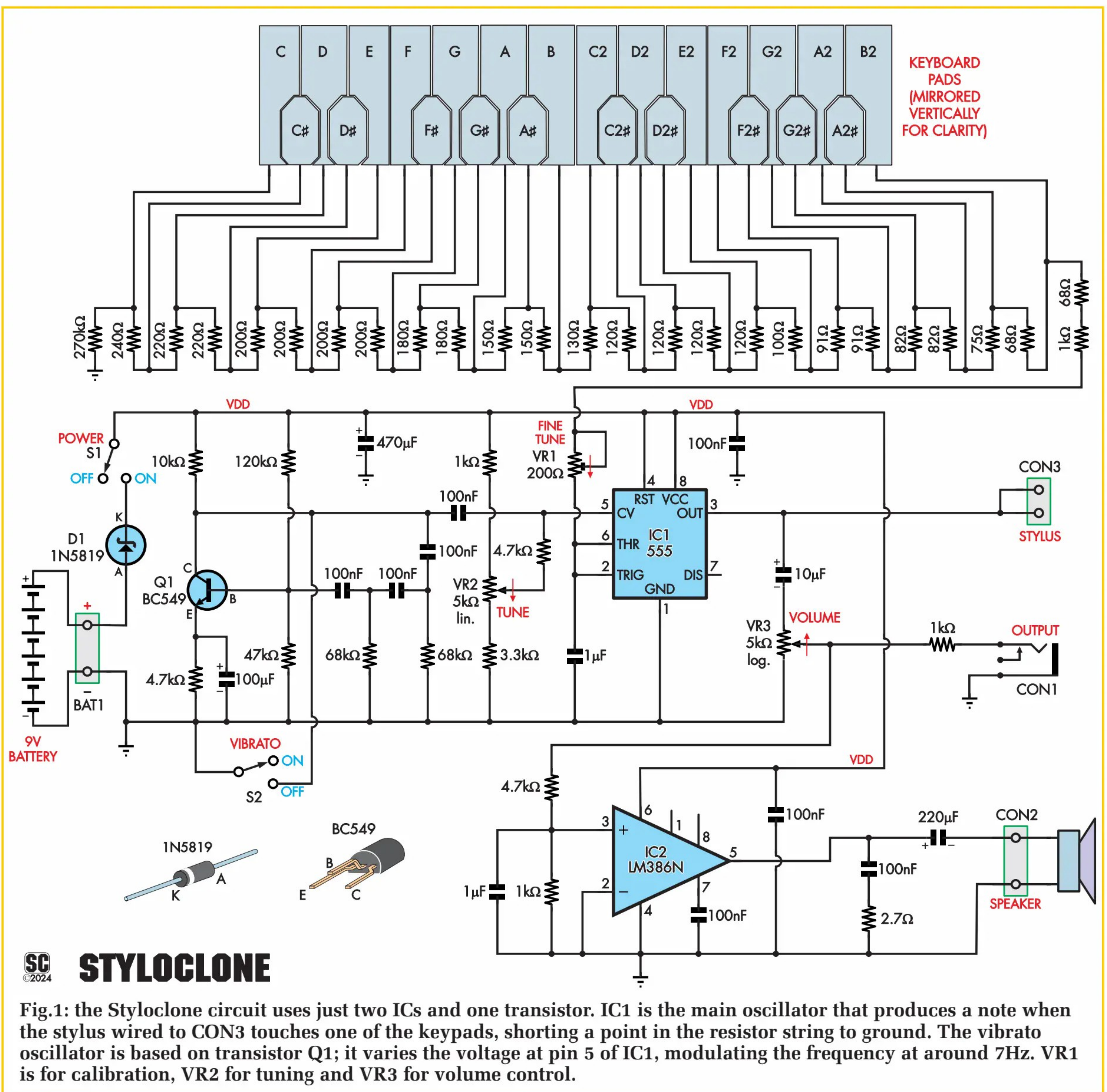
Ultimately, we decided to use the KISS principle ("keep it simple, stupid") and, if people enjoy the

old-school goodness of our take on a Stylophone, we could develop a new-fangled version later. Our Styloclone comprises a PCB, a stylus and an optional case, as shown in the photos.

The original Stylophone circuit has two main parts. The first is a simple oscillator in which the stylus changes the RC time constant to play the notes. The secondary vibrato oscillator causes the note frequency to vary slightly but rapidly, making the sound more interesting. You may recognise the concept of vibrato as it is applied to many instruments, including the human voice.

Early Stylophones used a unijunction transistor for the note oscillator but those are rare these days. Later versions used a 555 timer IC, which

Constructional Project



remains super common today, so we have also used one.

The original approach is a master-class in squeezing as much as possible from the minimum number of parts. We kept the essence and added a few new parts to make a modern, buildable project. That includes a simple output amplifier, allowing us to use a standard 8Ω speaker. The original circuit used a high-impedance speaker, which is now difficult to obtain.

We are using an LM386 amplifier, which is hardly dragging the Stylophone into the 21st century, as that part has been around since the mid-

1970s. The LM386 itself has some fame in the musical domain as a very common IC in guitar practice amplifiers and distortion boxes, so it is a fitting choice.

You can hear some audio clips of the prototype Styloclone at siliconchip.au/Shop/6/432 but remember that I am more of an engineer than a musician, so don't expect the Brandenburg Symphony! Still, they should give you an idea of the tone it produces if built as described here. It's possible to make some simple modifications to change the tone, some of which will be mentioned later.

Circuit details

The resulting circuit is shown in Fig.1. There are two oscillators, one for vibrato and the second to generate the notes. The vibrato oscillator is built around BC549 transistor Q1, with 100nF capacitors and 68kΩ resistors forming a feedback network. The result is a very simple phase-shift oscillator.

Early Stylophones used a 10MΩ pull-up resistor on the base of Q1 to bias this amplifier, which is effective but subject to significant variation. We have used a slightly more complex arrangement that ensures a defined setpoint for the

The Stylocclone



Scopes 1 & 2: the left scope shows the 1 μ F capacitor being charged and discharged by the 555 timer (cyan) and the output voltage being delivered to the speaker for a 440Hz A note (yellow). The control voltage (mauve) is about 5.3VDC in this case, although there is about 50mV AC superimposed on it from the output. The right scope shows the same signals as in Scope 1 but with a faster timebase and with vibrato enabled, visible as a periodic shifting of the waveform.

vibrato oscillator. It should work for any high-gain NPN transistor similar to the BC549.

The vibrato can be switched on and off using S2, which shorts the collector of Q1 to ground. This is a brutal but effective way of stopping this oscillator. We will discuss how the vibrato works as we describe the main oscillator.

The main oscillator in the original Stylophone used a programmable unijunction transistor in the main oscillator, although early updates replaced that with the NE555 timer IC, which came out in about 1972.

The 555 is set up as an astable multivibrator, which is a fancy way of saying that it oscillates continuously. For now, let's look at it with the vibrato switched off and the tuning potentiometer at its midpoint. Note that the 555 will work just fine without the tuning potentiometer but without the ability to adjust the tuning.

In this case, with the pen not touching any of the keyboard pads, both the trigger and threshold inputs are pulled to ground via trimpot VR1 and the long string of series resistors that ultimately connects to 0V. Referring to Fig.2, the output of Comparator C goes low, while Comparator T's goes high. The RS flip-flop is reset, so $\overline{\text{OUT}}$ goes low. The output buffer inverts this, so the 555 output goes high.

It remains in this state while no note is selected. When the stylus touches the keyboard, it connects the 555 output to part of the resistor string that defines each note. The 1 μ F timing capacitor charges via these resistors, with the charge rate determined by the resistance in the string (ie, the note selected).

It continues to charge until the voltage on the Threshold pin exceeds the Control voltage and the Comparator C output goes high. The RS flip-flop is reset, and the $\overline{\text{OUT}}$ pin goes high, so the 555 output pin goes low. The $1\mu\text{F}$ capacitor starts to discharge via the resistor string.

Once the voltage goes below the Comparator T positive input reference voltage, the Comparator T output goes high again, setting the RS flip flop. This drives the output high, and the whole process repeats. The resulting oscillation is demonstrated in Scope 1.

There are a few tweaks to the operation of the 555 IC in a Stylophone. The first is that the control voltage (CV) pin is connected to the wiper of a potentiometer. This varies the control volt-

age, which changes the voltage over which the $1\mu\text{F}$ capacitor must charge over the oscillation cycle, allowing the Stylocclone to be tuned.

We have selected resistors for each note that are in tune with middle A at 440Hz, as long as the 1 μ F capacitor is reasonably accurate. The tuning works well, but this is a very simple circuit, so if you set the tuning pot very high or low, you will find the octave is a bit off. We have selected values for the keyboard resistor string that give very close to in-tune notes.

The second tweak is that the vibrato oscillator is capacitively coupled to the 555's control voltage input. This adds an AC component to the control voltage and modulates the charge/discharge range required for the $1\mu\text{F}$ capacitor, as seen in Scope 2.

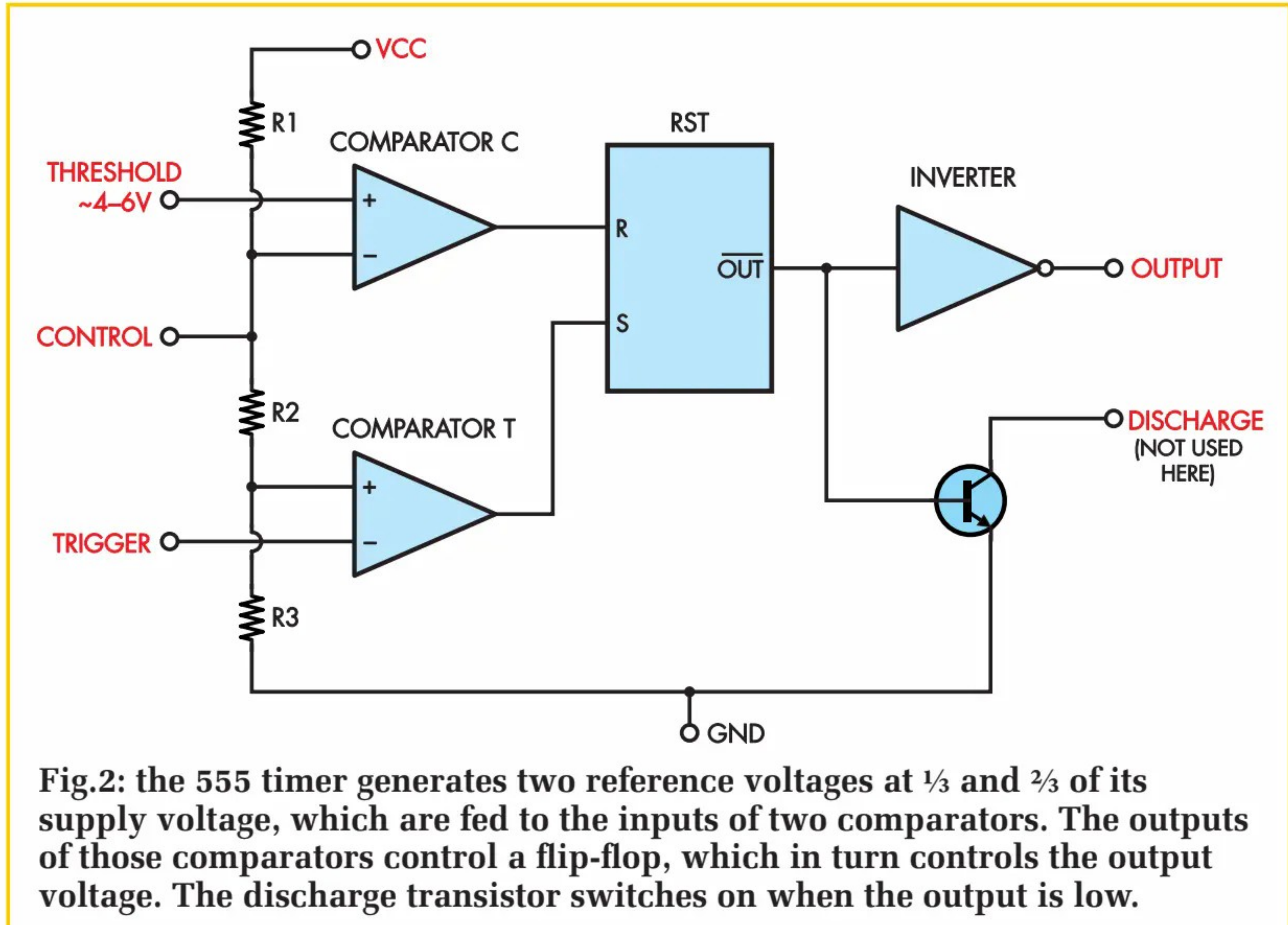
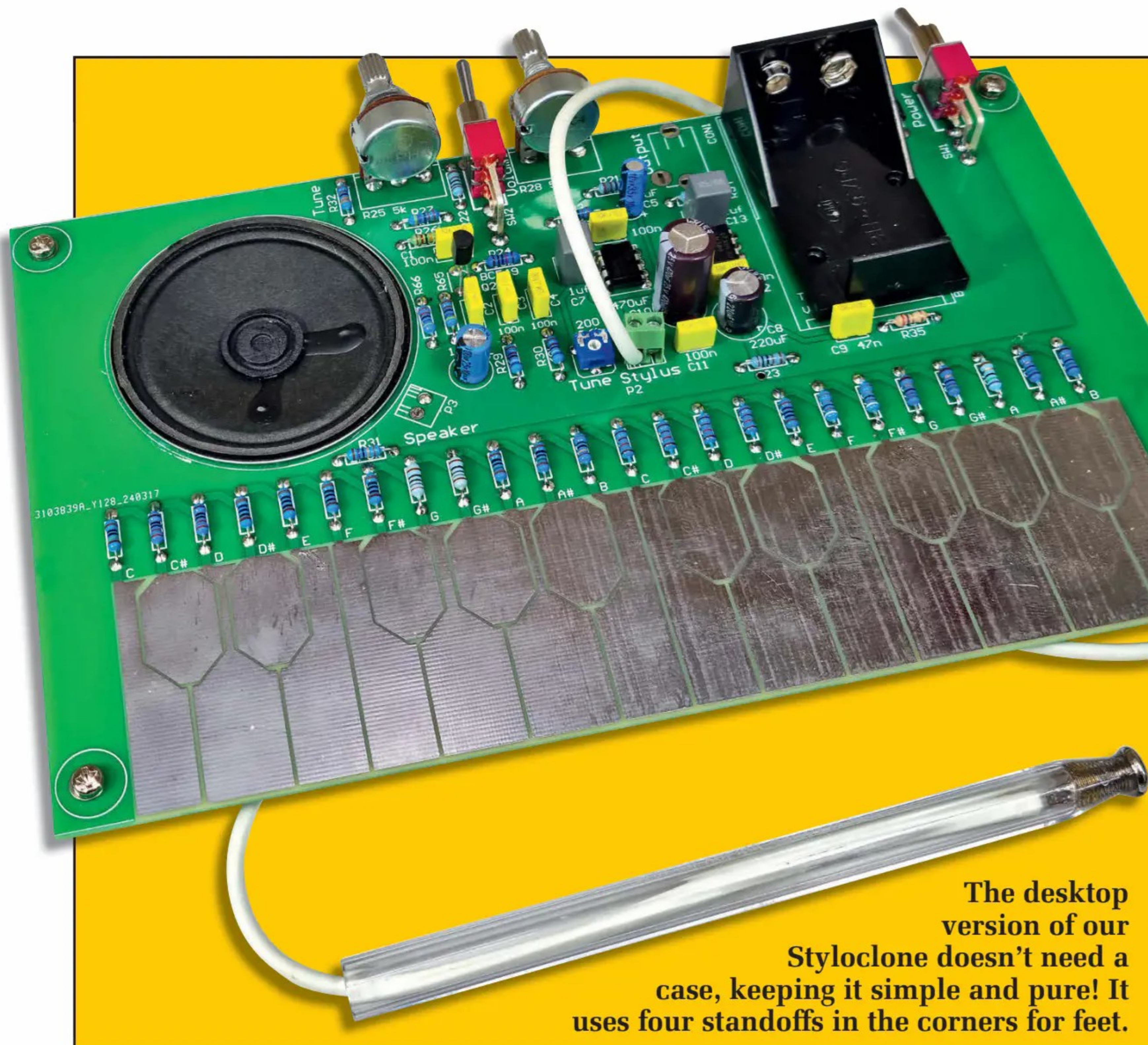


Fig.2: the 555 timer generates two reference voltages at $\frac{1}{3}$ and $\frac{2}{3}$ of its supply voltage, which are fed to the inputs of two comparators. The outputs of those comparators control a flip-flop, which in turn controls the output voltage. The discharge transistor switches on when the output is low.

Constructional Project



The desktop version of our Styloclone doesn't need a case, keeping it simple and pure! It uses four standoffs in the corners for feet.

Tuning calculations

Getting the notes right took a lot of work. The standard formula for oscillation frequency for a 555 timer is $f = 1.44 \div (C \times [R_a + 2 \times R_b])$. With the tweaks to the circuit, such as not using the discharge circuit, we found we needed to use $f = 0.5823 \div (R \times C)$. This is because we are not charging from V_{cc} and discharging to ground but instead using the 555 output as the charge/discharge source.

Even then, there was non-linearity across the scale; we were able to use this formula to get close to the right resistances, but then we had to hand-tune the values.

We were conscious that this might introduce variation in behaviour for chips from different suppliers, which may have differing high and low output voltages. To test this theory, we drove around town and bought five different chips from various suppliers and batches. We verified that the resistances we chose worked for all of these with only minor variations.

The fact that Stylophones have been made this way for years should have told us that we were jumping at shadows.

Our choice of a $1\mu F$ timing capacitor in the oscillator defined the resistances required for each note. Some rather

odd resistances are required. Table 1 shows each note's 'ideal' frequencies and incremental resistances. Because the resistance is in a string, we have worked out the best-fit values from the E24 resistance range.

As expected, there are minor errors. While these errors are not huge, they indicate there is no benefit in being overly anxious about achieving the exact modelled resistances. So, you can use 1% E24 resistors of the specified values; there is no need for more precision than that.

We considered having one potentiometer per note, but even the cheapest trimpots would have cost more than 10 times that of simple resistors. It would have also made tuning very complicated!

If you choose to fine-tune your Styloclone by varying the resistor values, remember to set the tuning control to your 'zero point' and keep it there while you select new resistors. You must start with the highest note and then work down the scale. All the tuning resistors add up, so if you go back and change a higher note, you need to retune all the lower notes.

We have lined all these resistors up alongside each note on the board. Make sure you check each value as you go and don't put any in the

wrong spot, or the tuning will end up all wonky.

The original Stylophone used a high-impedance speaker. We have added an amplifier and optional line output in case you want to record a hit song with your Styloclone. The circuit around the LM386 is bog standard, and the only part warranting comment is the $1\mu F$ capacitor from its pin 3 to ground that rolls off the high-frequency response.

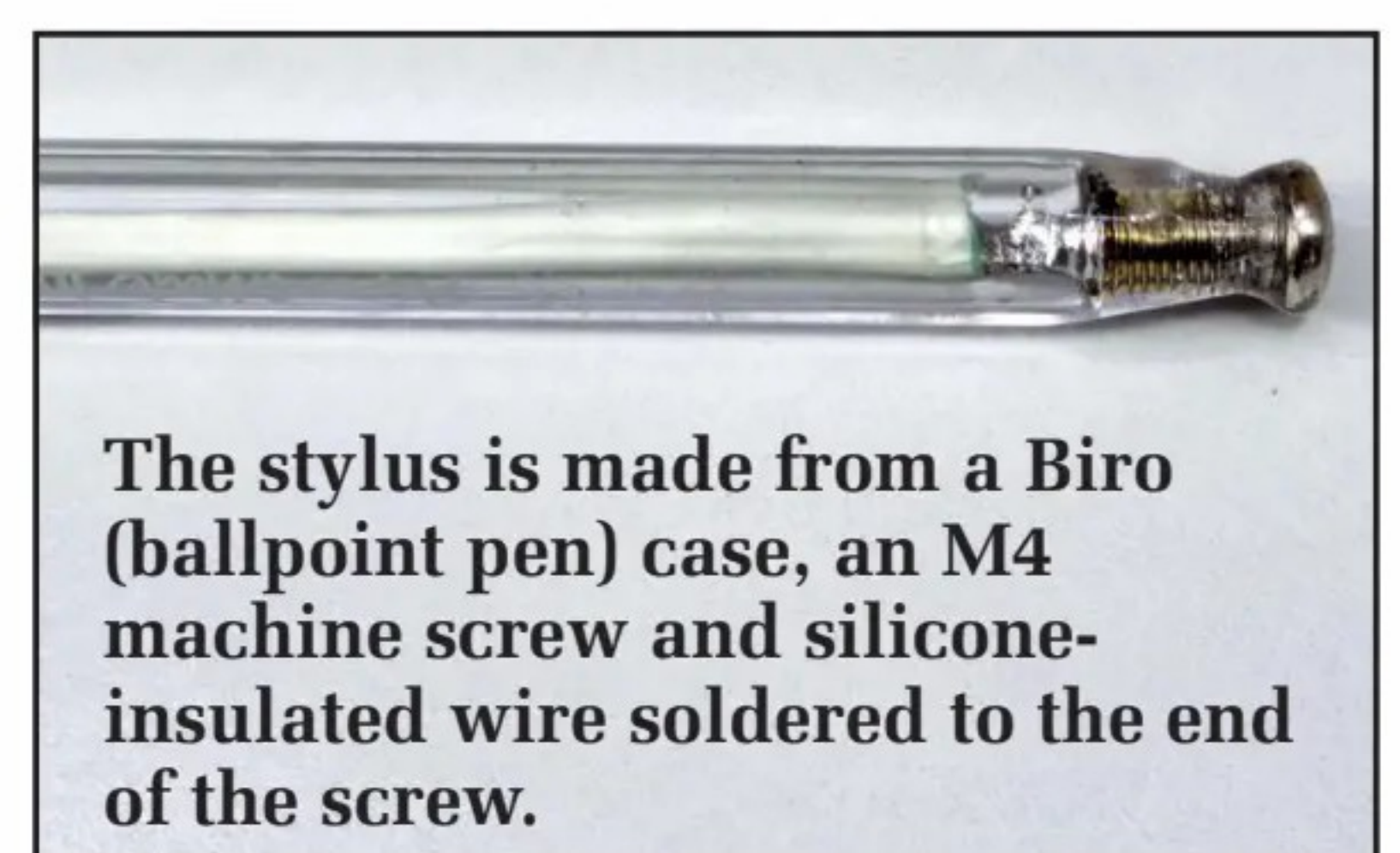
The resulting filter has a pretty brutal corner frequency of around 190Hz. The resistance of the RC circuit is formed by the $1k\Omega$ resistor in parallel with the $4.7k\Omega$ resistor from the volume control plus the volume control's resistance. This filter tones down the harshness of the square wave output a lot. If you want a brighter sound, reducing this $1\mu F$ capacitor will give you that.

We have used a simple 57mm speaker for this device; they are cheap and rugged. This speaker can produce plenty of output, but if more sound is required, you can certainly plug it into your Marshall Stack via the mono 3.5mm jack.

The stylus

We have used yet another Biro (ballpoint pen) case as the stylus handle in this project. This seems to be something of a tradition in the making! As the tip, we used a 4mm Posidriv machine screw (Phillips head would also be fine), to which we soldered the stylus lead. We then glued it into the tip of that obligatory Biro case using Araldite epoxy – see the photo below.

You might find another way of doing this. For example, you could simply place a small diameter heat-shrink tubing around a stiff piece of wire and bend the end back so it isn't sharp. However, if you use an alternative approach, ensure that the player is insulated from the stylus tip, as otherwise, skin resistance and body



The stylus is made from a Biro (ballpoint pen) case, an M4 machine screw and silicone-insulated wire soldered to the end of the screw.

The Styloclone

capacitance could interfere with its operation.

Our goal was to make a conductive stylus tip that did not have sharp edges that would scratch and wear the PCB ‘keys’.

For the stylus wire, we used super-flexible silicone-insulated wire on the assumption that the lead will be waved around a lot. We don’t want the lead breaking! The wire I used has 95 strands of copper with silicone insulation and is made for this sort of application (well, multimeter leads etc).

The large number of thin wires in the wire will make this very tolerant of flexing and maximises the fatigue life. The length of the stylus lead can be tweaked, but 600mm feels about right to us.

To attach the wire to the screw, we held the screw in a vise, applied flux to end top of the screw and tinned it. We then soldered the flexible wire to the end of the screw.

We have included two 4mm holes in the PCB to secure the stylus lead using a zip/cable tie in front of the stylus connector. This allows you to run the stylus lead through a simple hole you drill in the side of the case without the risk of it being pulled too hard and damaged. We recommend you select a side to suit your right- or left-handed preference.

Case options

We have produced two slightly different PCB designs. The first, coded 23106241, fits into a Gainta G1183 sloped instrument case and allows you to mount the board to the front panel using the inbuilt mounting holes. It gives a neat finish and delivers a neatly packaged product.

The way this board mounts requires all the components to be placed on the back of the board, with the ‘keys’ on the front of the board so they can be presented to the user through a large rectangular hole in the case.

However, we recognise that the case costs more than all the electronic components, and it isn’t strictly required. So the other PCB option, coded 23106242, has all the components on the top side of a rectangular board, with holes in the corners to use 10mm Nylon standoffs as feet. This way, you can set it on a flat surface to play it as a bare electronic assembly, or perhaps mount it onto

a piece of timber or other type of stiff board.

You need to decide which version you want to build before starting construction since the circuits are identical but the board layouts are quite different.

Case preparation

For the case-mounting version, we have put in some effort so that mounting the board is easy. The hardest part is neatly cutting the rectangular hole in the case to access the keys. We’ll describe how to prepare the case before assembling the PCB, as it might be easier to do it first. You can skip this section if you are building the version without the case.

The best approach to cutting the large hole is to mark its outline on the case, then drill 6mm holes in each corner 3mm inside the actual corner junction, so the edges of the holes align with the cutout. Next, use a hand saw or rotary tool to cut just inside the lines. You can then file the hole to size.

ABS plastic works very easily and does not clog files too severely, so tidying up the hole is a lot easier than you might expect. The drawing for the front panel cutout and drilling is in Fig.3.

I used a really sharp knife and ruler to score each line, but you have to be very careful not to slip and cut your fingers while doing that! If you cut like this repeatedly, you can actually go all the way through the plastic.

If you decide to do that, we suggest you wear chainmail (mesh) gloves. We are not making this up; chefs use them to avoid cutting their fingers. They are readily available, not too expensive, and surprisingly flexible. Search for “cut-resistant gloves” or “chef’s gloves” to find them.

The remainder of the case preparation is drilling the speaker holes on the top panel, plus the switch and potentiometer holes in the rear panel.

We found it kind of fiddly to get the measurements right for the rear panel, which is at an angle to the front panel and has rounded edges. So be

Table 1 – Styloclone note ideal frequencies, resistors & actual frequencies

Note	Ideal	Resistor	Running total	Measured	Error	Error (%)
B	493.9Hz	68Ω	1179Ω	493Hz	-0.9Hz	-0.3 to -0.1
A#	466.2Hz	68Ω	1247Ω	466Hz	-0.2Hz	-0.2 to +0.1
A	440.0Hz	75Ω	1322Ω	440Hz	0.0Hz	-0.1 to +0.1
G#	415.3Hz	82Ω	1404Ω	416Hz	0.7Hz	+0.0 to +0.3
G	392.0Hz	82Ω	1486Ω	394Hz	2.0Hz	+0.4 to +0.6
F#	370.0Hz	91Ω	1577Ω	371Hz	1.0Hz	+0.1 to +0.4
F	349.2Hz	91Ω	1668Ω	351Hz	1.8Hz	+0.4 to +0.7
E	329.6Hz	100Ω	1768Ω	332Hz	2.4Hz	+0.6 to +0.9
D#	311.1Hz	120Ω	1888Ω	312Hz	0.9Hz	+0.1 to +0.5
D	293.7Hz	120Ω	2008Ω	294Hz	0.3Hz	-0.1 to +0.3
C#	277.2Hz	120Ω	2128Ω	278Hz	0.8Hz	+0.1 to +0.5
C	261.6Hz	120Ω	2248Ω	264Hz	2.4Hz	+0.7 to +1.1
B	246.9Hz	130Ω	2378Ω	251Hz	4.1Hz	+1.5 to +1.9
A#	233.1Hz	150Ω	2528Ω	236Hz	2.9Hz	+1.0 to +1.5
A	220.0Hz	150Ω	2678Ω	223Hz	3.0Hz	+1.1 to +1.6
G#	207.7Hz	180Ω	2858Ω	210Hz	2.3Hz	+0.9 to +1.3
G	196.0Hz	180Ω	3038Ω	198Hz	2.0Hz	+0.8 to +1.3
F#	185.0Hz	200Ω	3238Ω	186Hz	1.0Hz	+0.3 to +0.8
F	174.6Hz	200Ω	3438Ω	175Hz	0.4Hz	-0.1 to +0.5
E	164.8Hz	200Ω	3638Ω	164Hz	-0.8Hz	-0.8 to -0.2
D#	155.6Hz	200Ω	3838Ω	155Hz	-0.6Hz	-0.7 to -0.1
D	146.8Hz	220Ω	4058Ω	147Hz	0.2Hz	-0.2 to +0.5
C#	138.6Hz	220Ω	4278Ω	139Hz	0.4Hz	-0.1 to +0.6
C	130.8Hz	240Ω	4518Ω	131Hz	0.2Hz	-0.2 to +0.5

The Error column has an uncertainty of ±0.5Hz.

Constructional Project

cautious with this; perhaps start with smaller holes than required and be prepared to file them to a final size. Fig.4 shows the drilling details for the rear panel.

Regarding labelling, we were on a roll with the retro feel of this project and had just purchased a 3D printer for the young enthusiast. So we got out the 3D modelling software and made a cool label for the project. We reckon it looks pretty good. The STL file is available from siliconchip.au/Shop/11/434

Note that it needs to be printed at a 5% scale, a simple selection in the Cura slicer program.

You could probably use super glue to attach the PLA 3D-printed parts to the ABS plastic case, but we read that you can melt ABS plastic in acetone to make “ABS glue”. We picked up some of the swarf from drilling the front panel, put it in a teaspoon of ac-

etone and mixed until we had a thick, sticky liquid.

We dabbed it onto the back of the PLA labelling and carefully placed it on the front panel, where it stuck perfectly. Get it in the right spot when you put it down and do not move it.

PCB assembly

Building the Styloclone electronics is pretty straightforward. All parts are through-hole types, and we have used larger pads where possible to facilitate soldering.

First, check that you have the correct PCB, either the one coded 23106241 that measures $179 \times 123\text{mm}$ for the case-mounting version or the standalone board that's coded 23106242 and measures $207 \times 124.5\text{mm}$. Figs.5 & 6 are the PCB overlay diagrams for the two versions that show where all the components go.

The best place to start is with the resistors. We have had to use several E24 resistors with less-familiar values like 91Ω , 200Ω etc. These are used to get the tuning right for each note, so you really need the specified parts. Most local retailers stock these E24 values, and the larger online suppliers like Farnell, Mouser, DigiKey and RS have them too.

When fitting these, we recommend measuring each part's resistance as you go since the colour codes can be tricky to read sometimes. If you get a part in the wrong spot, you will find that some of the notes are out of tune. Once all the resistors are in place, add 1N5819 schottky diode D1 (taking care to match its orientation to the overlay) and the 200Ω trimpot.

At this point, it is convenient to mount the 555 and LM386 ICs. Do this before the capacitors, as it will give

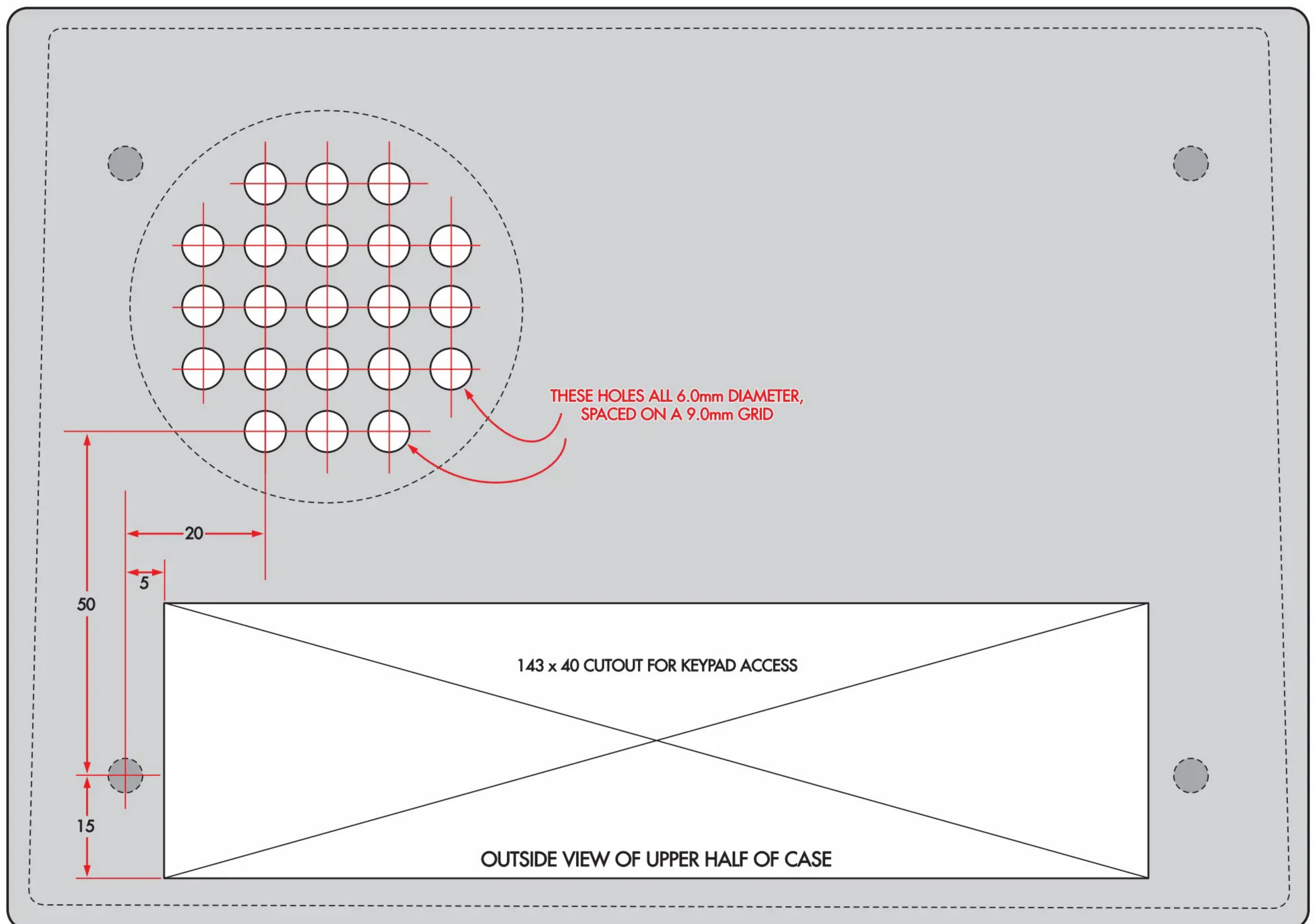


Fig.3: cutting the large rectangular hole is the fiddliest part of the project. We used a Dremel cutting tool and file for ours. The easiest way is to mark and drill the corner holes for the cutout from the inside, then do the remainder of cutting and drilling from the outside. If you need an inside template, you can print this out mirrored. Regardless, double-check which side you drill the speaker holes on. All dimensions in this diagram are in millimetres.

The Styloclone



you more room to get them in place. The 555 and LM386 look the same, so you will need to check the part numbers and ensure they are both the right way around. The dot or indent on the chips goes to the top of the board, and the PCB silkscreen has dots in nearby positions to help you.

It is then time to install the capacitors. Make sure you use film capacitors for values up to 1 μ F; either MKT or greencaps will work fine, although greencaps may need their leads bent to fit the pads. Do not use ceramic capacitors, as they have a huge temperature coefficient and large values can even be microphonic.

The 1 μ F capacitor on pin 2 of the 555 is particularly critical; it must be close to 1 μ F, so use a part with a

decent tolerance (5% if possible; failing that, 10%).

After that, you can add the electrolytic capacitors. Make sure each one is the right way around, with the longer positive lead on the side with the + symbol. The stripe on the can indicates the negative side, so it should be opposite the + symbol in each case. To make this process easier, all capacitors are orientated in the same direction.

Next come the two 5mm screw terminals and the battery clip. We like to make connections to offboard components using terminals as it makes it easy to service, but you could simply solder flying leads to the board if that suits you. Put a dab of neutral-cure silicone sealant under the battery clip before you solder it to the board, as

that will keep it secure when the Styloclone is in use.

We have added two 4mm holes on either side of the battery, allowing you to 'zip tie' it to the board so it can't come out during transport.

Now fit the two switches. We have used PCB-mounting switches from a local supplier. Similar switches are available from Mouser and other larger suppliers; you could run flying leads from the PCB pads to panel-mounted switches in a pinch.

Two similar but not identical types of potentiometers are used: a 5k Ω linear type for the tuning control and a 5k Ω logarithmic type for the volume control. Both are standard 16mm-size devices. The volume pot will be labelled A5K, where A indicates "Audio

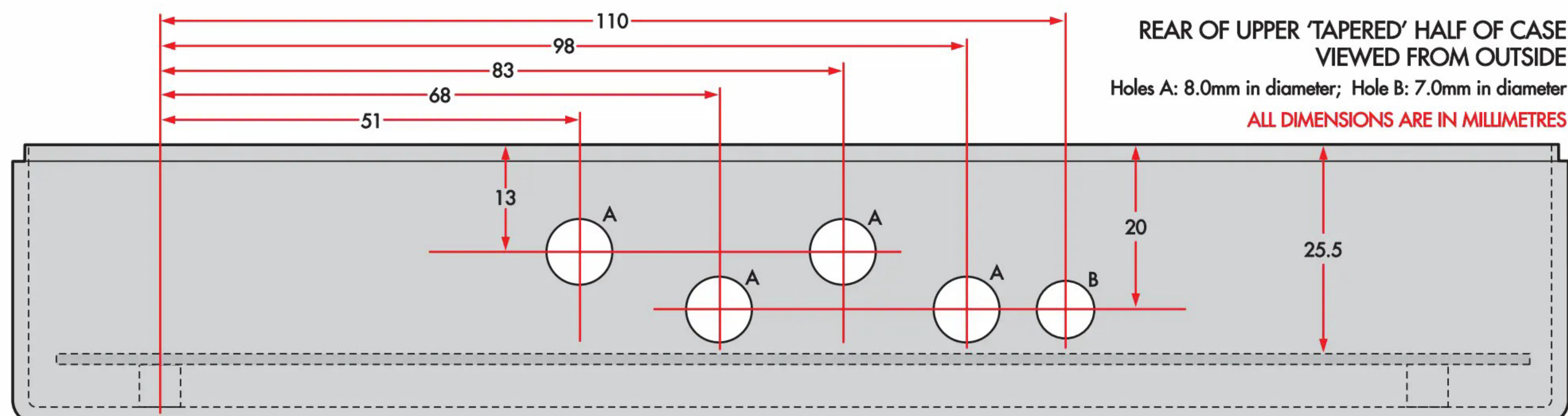
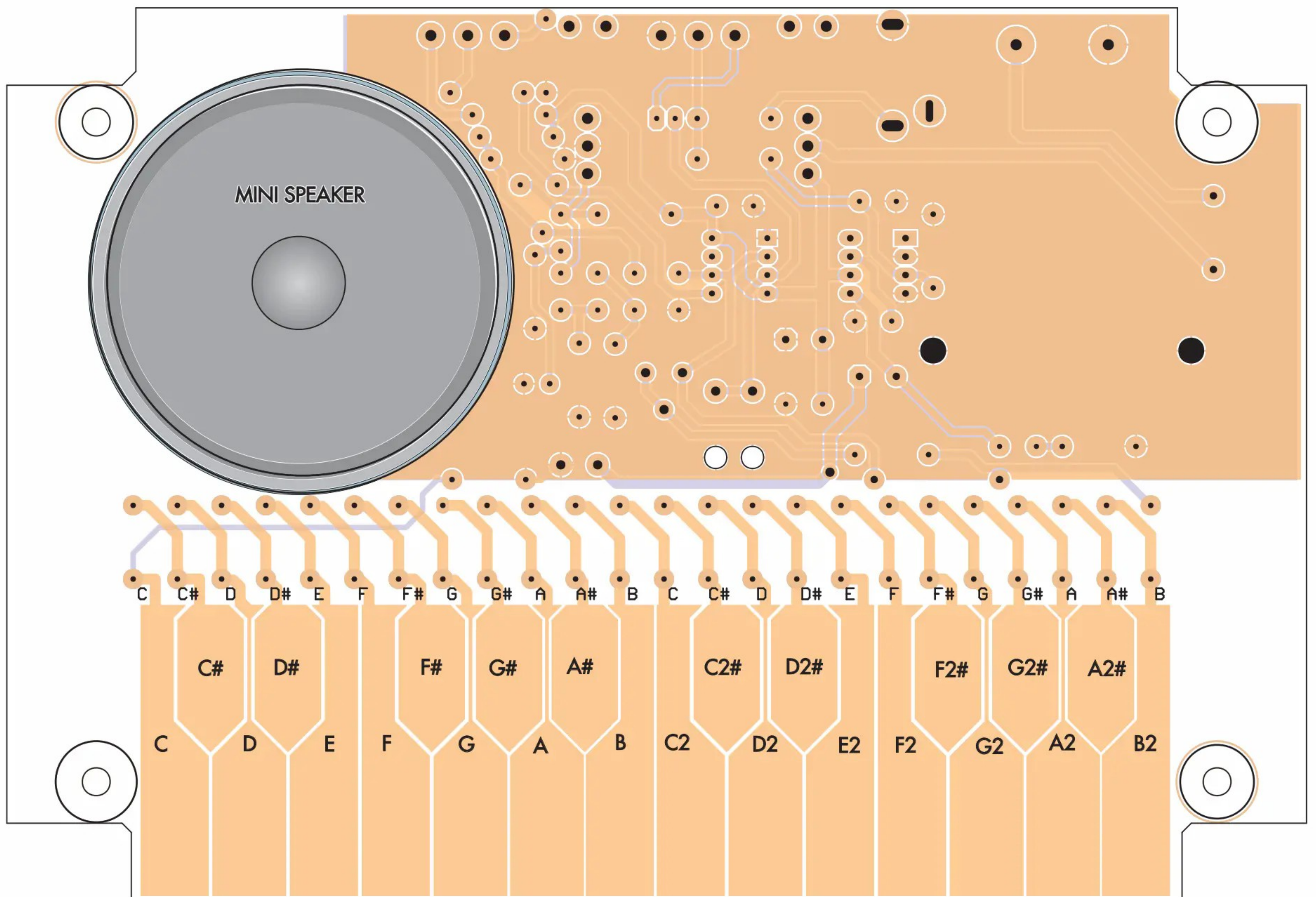
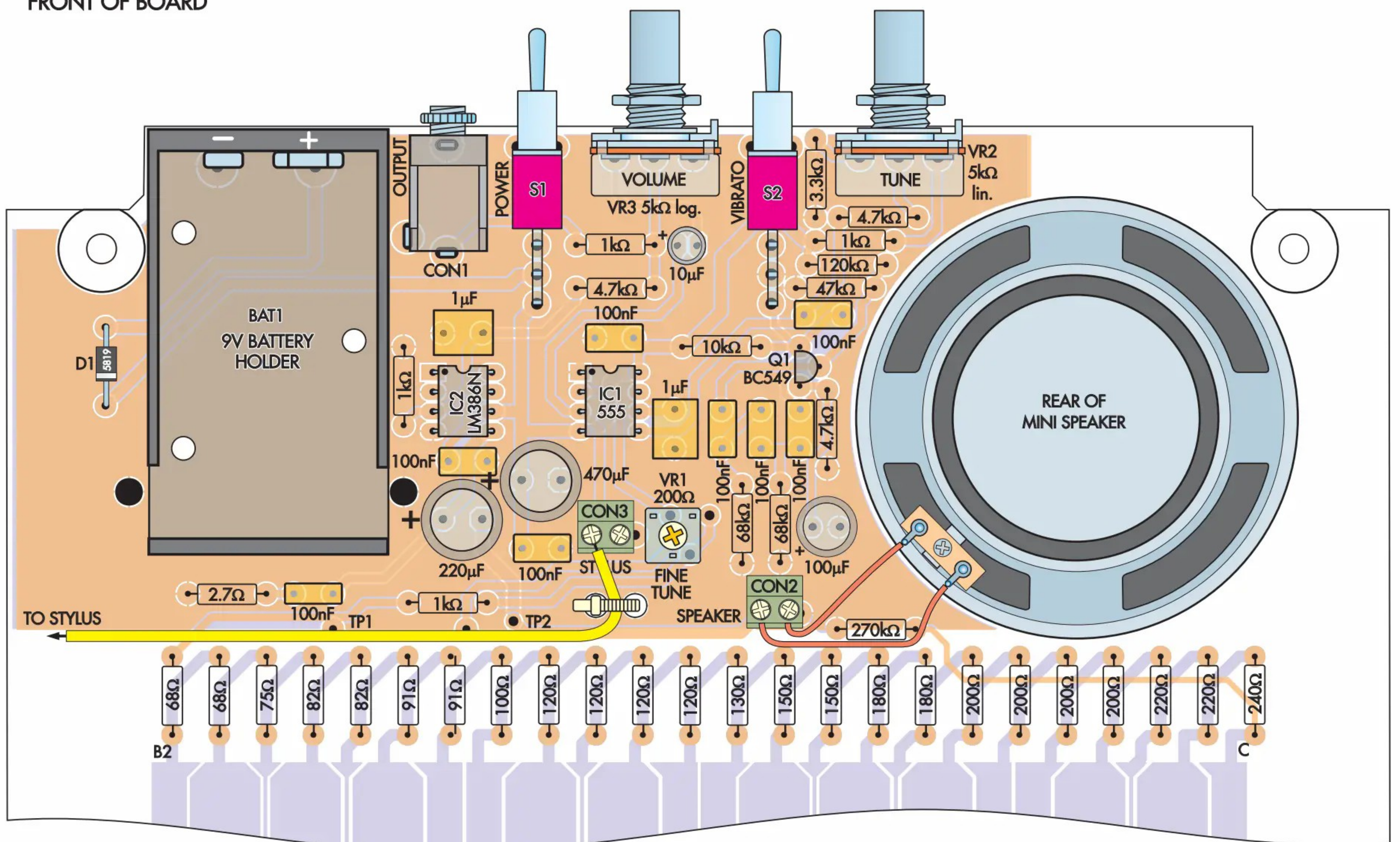


Fig.4: the drilling details for the Styloclone's rear panel. This is the top of the case; the front panel is at the bottom in this drawing. All measurements are referenced to the top of the fixing post on the left.

Constructional Project



FRONT OF BOARD



UNDERSIDE OF BOARD (TRUNCATED)

The Stylocclone

Taper”, while the tuning pot is linear and will be labelled “B5K”. Do not get these mixed up.

If you intend to plug the Stylocclone into an amplifier or recorder, mount the 3.5mm socket now. If you will never use it, you can save yourself a bit of money and leave it off.

We used a thin bead of neutral-cure silicone sealant around the hole in the PCB to attach the speaker. After applying the sealant, gently push the speaker into place. An alternative is to use 5-minute Araldite (or another epoxy glue), which works a treat and is pretty permanent. The speaker must go in so that the cone is visible from the same side as the tin-plated 'keys'.

If building the board designed to mount in the case, the speaker will be inserted from the opposite side of the board to the majority of the components, so the magnet comes through to the same side as the components and the cone can present through the front panel.

If you are building the standalone board, the speaker is inserted from the same side as the components, and the magnet will be on the underside of the board.

Once you glue the speaker in, take a break and ensure your glue cures. Once that silicone cures, the speaker will never fall out, but until then, it will fall out and make a mess of everything near it. Don't ask me how I know!

Solder wires to the speaker terminal and screw them into the speaker header; it doesn't matter which way around they go. Also connect a wire to the stylus connector for testing. Depending on whether you are left- or right-handed, drill a 3mm hole for your stylus wire to go through in one side of the case. This should be on the top half of the case, about halfway down the length.

Testing your Stylocclone

Once all the parts are in, insert a fresh battery and measure the voltage across it. It should be near 9V. If the voltage is low, look for parts getting hot; if none are, pull the battery out and check that it is fresh.

With the voltage rail all good, set the volume and tuning controls to mid-range and touch the stylus to one of the key pads. You should hear a tone. Run up and down the keys and you should get a reasonable set of notes. If you get nothing:

Fig.5: here is where to fit all the components for the case-mounting version of the Stylocclone. In this version, all parts mount on the bottom while the 'keys' are on the top. Take care with the orientations of the diode, ICs and electrolytic capacitors and make sure you don't mix up all the different-value resistors!

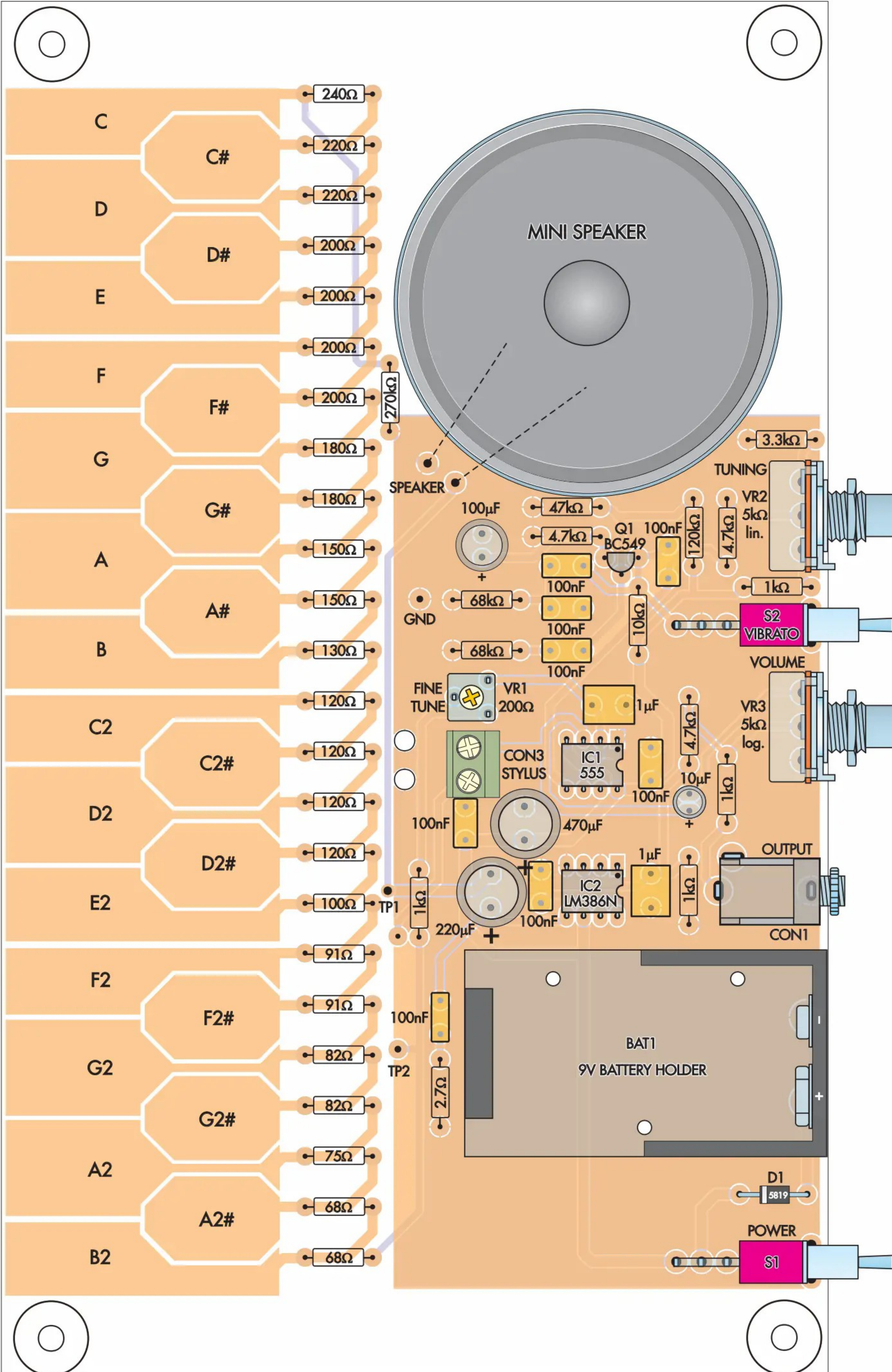


Fig.6: for the standalone (non-case) version of the Stylophone, all parts mount on the top side, which also has the keyboard. All the same parts are used in both versions. Again, watch the orientations of the diode, ICs and electrolytic capacitors and make sure you don't mix up all the different-value resistors.

Constructional Project

❑ Check the voltage on pin 8 of IC1, the 555 timer. It should be more than 8V. If not, then something is wrong with the power supply. Is it on? Is diode D1 the right way around?

❑ Is IC1 indeed a 555 and is it the right way around?

❑ Check that pin 5 of IC1 is between 4V and 6V. The tuning pot sets this, so try adjusting that. It does not need to be exact, but it should not be pegged to one of the rails.

❑ Check for an AC voltage on pin 3 of IC1. This will be a square wave.

❑ If there is a voltage there, trace through the 10µF capacitor to the clockwise terminal of volume control VR3, then to its wiper and on to pin 3 of IC2, the LM386. Try turning the volume up if you lose the signal at the wiper of VR3.

❑ Check pin 6 of IC2 for 8-9V DC. If this is not present, track back to the battery again.

❑ Check pin 5 of IC2 for an AC voltage. If this is present, is the output capacitor the right way around, and is the speaker wired up properly? Are its terminals shorted?

If the notes are all wrong:

❑ Check that 200Ω trimpot VR1 on the board is set to around 110Ω. You should measure close to 1110Ω across test points TP1 and TP2, which are just below the battery (this measures VR1 and a 1kΩ series resistor).

❑ Check that the correct values have been used for the row of resistors near the keys.

❑ Set the tuning potentiometer, VR2, to about 2/3 scale. Turn the Styloclone on and measure the voltage at pin 5 of the 555 timer. This should be about 5.3V. If not, adjust the trimpot and see if it can be set to about 5.3V. Check that you have not swapped the linear and log pots.

❑ Touch the stylus to the high B. You should get a reasonably high note at around 494Hz. If this is way off, check the value of the 1µF timing capacitor. If this note is wrong, every other note will also be wrong.

❑ Assuming the high B is OK, run along the notes going down the scale. If you find a wrong note, check the associated resistor and correct the problem. Repeat until they are all correct.

❑ Remember that all lower notes are built on the preceding notes, so you should only fix a resistor associ-

Parts List – Styloclone

- 1 134 × 189 × 55mm sloping ABS desktop instrument case
[Gainta G1183B(UL), available from TME] OR
- 4 M3 × 10mm tapped Nylon spacers
- 1 double-sided PCB coded 23106241 (case version), 179 × 123mm **OR**
- 1 double-sided PCB coded 23106242 (standalone version), 207 × 124.5mm
- 1 57mm (2¼-inch) diameter 8Ω 700mW+ loudspeaker (SPK1)
- 2 PCB-mount right-angle miniature SPDT toggle switches (S1, S2)
- 1 200Ω top-adjust mini trimpot (VR1)
- 1 5kΩ 16mm single-gang linear (B5K) potentiometer (VR2)
- 1 5kΩ 16mm single-gang logarithmic (A5K) potentiometer (VR3)
- 1 PCB-mount 9V battery holder (BAT1)
- 1 PCB-mount 3.5mm SPST 3-pin mono jack socket (CON1; **optional**)
[SCHURTER 4832.222, Farnell 152206]
- 2 2-way 5mm/5.08 miniature PCB-mounting terminal blocks (CON2, CON3)
- 4 M3 × 6mm panhead machine screws
- 4 M3 shakeproof (star) washers
- 1 ballpoint pen case
- 1 short M4 panhead machine screw
- 1 9V battery
- 2 short 2.5mm- or 3.5mm-wide cable ties ('zip ties')
- 1 60cm length of white silicone-insulated hookup wire (outside diameter ~2.5mm)

Semiconductors

- 1 555 timer IC, DIP-8 (IC1)
- 1 LM368N mono amplifier IC, DIP-8 (IC2)
- 1 BC549 30V 100mA NPN transistor, TO-92 (Q1)
- 1 1N5819 40V 1A schottky diode (D1)

Capacitors (16V electrolytic unless noted)

- 1 470µF 1 10µF 50V electrolytic
- 1 220µF 2 1µF ±5% 63V/100V MKT
- 1 100µF 8 100nF 63V/100V MKT

Resistors (all ¼W 1% axial unless noted)

- | | | |
|---------|--------|----------------|
| 1 270kΩ | 4 1kΩ | 4 120Ω |
| 1 120kΩ | 1 240Ω | 1 100Ω |
| 2 68kΩ | 2 220Ω | 2 91Ω |
| 1 47kΩ | 4 200Ω | 2 82Ω |
| 1 10kΩ | 2 180Ω | 1 75Ω |
| 3 4.7kΩ | 2 150Ω | 2 68Ω |
| 1 3.3kΩ | 1 130Ω | 1 2.7Ω (5% OK) |

Value	4-band code	5-band code
2.7Ω		
68Ω		
75Ω		
82Ω		
91Ω		
100Ω		
120Ω		
130Ω		
150Ω		
180Ω		
200Ω		
220Ω		
240Ω		
1.0kΩ		
3.3kΩ		
4.7kΩ		
10kΩ		
47kΩ		
68kΩ		
120kΩ		
270kΩ		

ated with a wrong note if all the notes above it are correct.

Switch on the vibrato using switch S2 and check that you get a warbling effect. Now you should have a working Styloclone.

We used VR1 and VR2 to tune the upper A on our unit to 440Hz. At this frequency, the resistors we selected have pretty much the whole range of notes in tune (just as importantly, you'll be in tune with a concert grand piano). The tuning process is:

1. Using a DMM, adjust trimpot VR1 to achieve 1110Ω between TP1 and TP2.

2. Adjust tuning potentiometer VR2 to get 440Hz at pin 3 of IC1 when

the upper A is played. If you have a frequency meter, probe the speaker output.

3. Check that the other notes are in tune. If they are not, use a DMM to check the associated resistor values.

How long will the battery last? That depends on how loud you play it. When switched on but not playing a note, ours drew 8.5mA. A typical 9V battery would idle for about 50 hours before going flat. At moderate volumes, the current draw increases to about 60mA, which means it should provide about 3-6 hours of playing time.

You should now be set to go and create your masterpiece! **PE**

The History of Electronics *Inventors and their Inventions*

Over the last five issues, we have described many individuals and organisations that made vital contributions to electronics. Their work made modern technology possible. Many significant developments also emerged from universities, companies and other organisations since 1962, described in this sixth and ultimate part of the series.

Part 6: by Dr David Maddison

Image Source: www.pexels.com/photo/2047905/

This final article covers significant inventions that cannot be attributed to an individual made from 1963 to the present. They are associated with a company or governmental organisation, either because we don't know the name(s) of the inventors, or because they were part of a team. Unlike the first four parts, which were organised by the date of birth of the inventor, this one (like the last) will list them by the year of the invention or discovery.

Because this list is of more recent inventions, you will notice that most of the entries are related to developments in computers, communications and software.

Of course, there are too many technological developments over the last 62 years to list them all here; this will merely be a sampling of what we think are the most significant developments.

Philips compact cassette, ASCII etc

1963

COMPAC (the Commonwealth Pacific Cable System) undersea telephone

cable was completed, linking Australia, New Zealand and Canada via Hawaii and Fiji. Parts had been operating since 1961. This coaxial cable could handle 80 phone calls or 1760 teleprinter circuits. It replaced HF radio telephone calls, which had to be booked and were delayed if transmission conditions were bad.

Philips introduced the first audio cassette tape, the "Compact Cassette".

The first transpacific television transmission via satellite was made between



1591 ABS flame-retardant enclosures

Learn more:
hammondmfg.com/1591

uksales@hammfg.com • 01256 812812



Feature Article

Japan and the USA, via the experimental Relay 1 communications satellite in an elliptical orbit.

Nottingham Electric Valve Company in the UK released the Telcan (Fig.68), a videotape recorder intended for domestic use. It used ¼-inch audio reel-to-reel tape running at 305cm per second, a very high speed for this type of tape, and could record up to 20 minutes of monochrome video on one of two tracks. The recording bandwidth of 2.6MHz provided 405 lines.

It was mainly sold as a kit, for £60, equivalent to about £1000 today. It was a commercial failure; for more details, see <https://pemag.au/link/abp1>

The first edition of the ASCII character encoding standard was published.

| TPC-1, Xerox fax system, BASIC etc **1964**

The Trans-Pacific cable system, TPC-1, linking Japan, Guam, Hawaii and the mainland USA became operational. It carried 128 telephone circuits.

Xerox introduced the first modern commercial fax system, which they called Long Distance Xerography or LDX.

The BASIC computer programming language was released.

The first prototype Moog electronic music synthesiser was built by Robert Moog (1934-2005). Commercial models were produced from 1967.

| geosynchronous satellite Intelsat I etc **1965**

The Dadda hardware binary multiplier was invented by Luigi Dadda (1923-2012) for computer arithmetic operations. It was smaller and faster than the previous implementation.

Sony released the CV-2000 (CV = "consumer video"), the first mass-produced domestic video tape recorder

(see Fig.69). It recorded in monochrome and used 13mm tape in a reel-to-reel format. It had broad uptake among business and educational institutions. Its inability to adjust head tracking meant it was impossible to swap tapes between machines; that was corrected in later versions.

The first commercial geosynchronous satellite, Intelsat I, was launched. It carried either 240 telephone circuits or one TV circuit. It was in use for over four years until it was deactivated, with a temporary reactivation for use for the Apollo 11 mission, and another temporary reactivation in 1990 to mark its 25th Anniversary. It is still in orbit.

| Magnafax telecopier (fax machine) **1966**

Xerox introduced the first easy-to-use fax machine, the Magnafax Telecopier, that used standard telephone lines.

| PAL standard, ATM, WRESAT etc **1967**

The SECAM colour television standard was released and adopted in France.

PAL standard colour television started broadcasting in the UK.

The world's first automated teller machine (ATM) was installed at Barclay's Bank, Enfield, UK. It was operated by inserting cheques, previously issued by a teller, marked with radioactive carbon-14 for machine readability and to confirm their authenticity.

Australia's first locally made satellite, WRESAT, was launched. SILICON CHIP had an article on WRESAT in its October 2017 issue (see siliconchip.au/Article/10822).

| LCDs, Group 1 fax standards **1968**

A team at RCA Laboratories demonstrated an 18×2 matrix liquid crystal display (LCD) using dynamic scattering mode (DSM), invented by George

Heilmeyer (described in the April issue).

The ITU (International Telecommunications Union) released Group 1 fax standards. Conforming machines took about six minutes to transmit a page at 96 lines per inch (38 per cm).

| MOS DRAM, Unix, ARPANET etc **1969**

Commercial production of MOS DRAM (metal oxide semiconductor dynamic random access memory) was started by Advanced Memory Systems, Inc, and was offered to selected companies. The chips contained 1024 bits of memory.

In the same year, Intel produced the 1103 memory chip, also with 1024 bits, and sold it on the open market. It was used in popular computers such as the HP 9800 series and the PDP-11.

The Unix operating system for computers was released.

The first commercial quartz oscillator watch was introduced, the Seiko Quartz-Astron 35SQ. It had an accuracy of ±5 seconds per month and a battery life of around one year.

The US Department of Defense (DoD) Advanced Research Projects Agency (ARPA) established a packet-switched computer network, ARPANET (see Fig.70), which eventually evolved into the internet we have today.

| digital fax machine, pocket calculator **1970**

Dacom produced the first digital fax machine, the DFC-10, that used data compression and could transmit a page in under one minute.

The Pascal computer programming language was released.

The first commercial handheld pocket calculator, the Canon Pock-etriconic (Fig.71), became available. It was influenced by the prototype



Figs.68: a Telcan home video recorder, sold mainly as a kit using ¼-inch audio tape. Source: www.nottinghampost.com/news/history/20-best-things-nottingham-given-192680



Fig.69: the Sony CV-2000, the first mass-manufactured video recorder for the domestic market. It used half-inch (12.7mm) reel-to-reel tape. Source: www.smecc.org/sony_cv_series_video.htm

The History of Electronics, part 6

Texas Instruments Cal Tech calculator of 1967 and used three TI MOS integrated circuits. It had no display; results were printed on paper tape. For more information, see <https://pemag.au/link/abp4>

| Intel 4004, Kenbak-1 PC, EPROM etc **1971**

The first commercial microprocessor, the Intel 4004, was released. It was mainly intended for calculators and cash machines.

The US DoD funded a five-year program to make a speech recognition machine that could recognise 1000 words within sentences. A machine called Harpy was built that recognised 1,011 words; see the PDF at <https://pemag.au/link/abp5>

Docutel introduced the "Total Teller" machine, an ATM that could accept deposits, transfer from one account to another and dispense cash. It operated offline using plastic cards and had a mechanical display with messages on a printed cylinder. By 1975, 3000 ATMs had been installed worldwide, 80% from Docutel. In 1982, Docutel merged with Olivetti.

The first personal computer (without a microprocessor) was released, the Kenbak-1 (Fig.72). Only 40 were sold.

Intel released the first EPROM (Erasable Programmable Read Only Memory), invented by Dov Frohman. The Intel 1702 could store 256 bytes of data.

Sony released the U-matic video cassette format to market, the first commercial video cassette format. It used 3/4-inch (19mm) tape. It was initially intended for the consumer market but was too expensive; it became popular in the institutional and industrial markets, plus the television industry.

SILICON CHIP magazine had a detailed series of articles on the history of videotape recording in its March, April, May & June 2021 issues (see siliconchip.au/Series/359).

| Intel 8008, C, Pong, blue LEDs etc **1972**

The Philips VCR (video cassette recorder) format N1500 player/recorder was introduced for the domestic market. The last Philips VCR recorder was released in 1979.

The eight-inch (20.3cm) floppy disk was commercially released.

The ITU (International Telecommunications Union) Group 2 fax standards were published. Conforming machines took about three minutes to

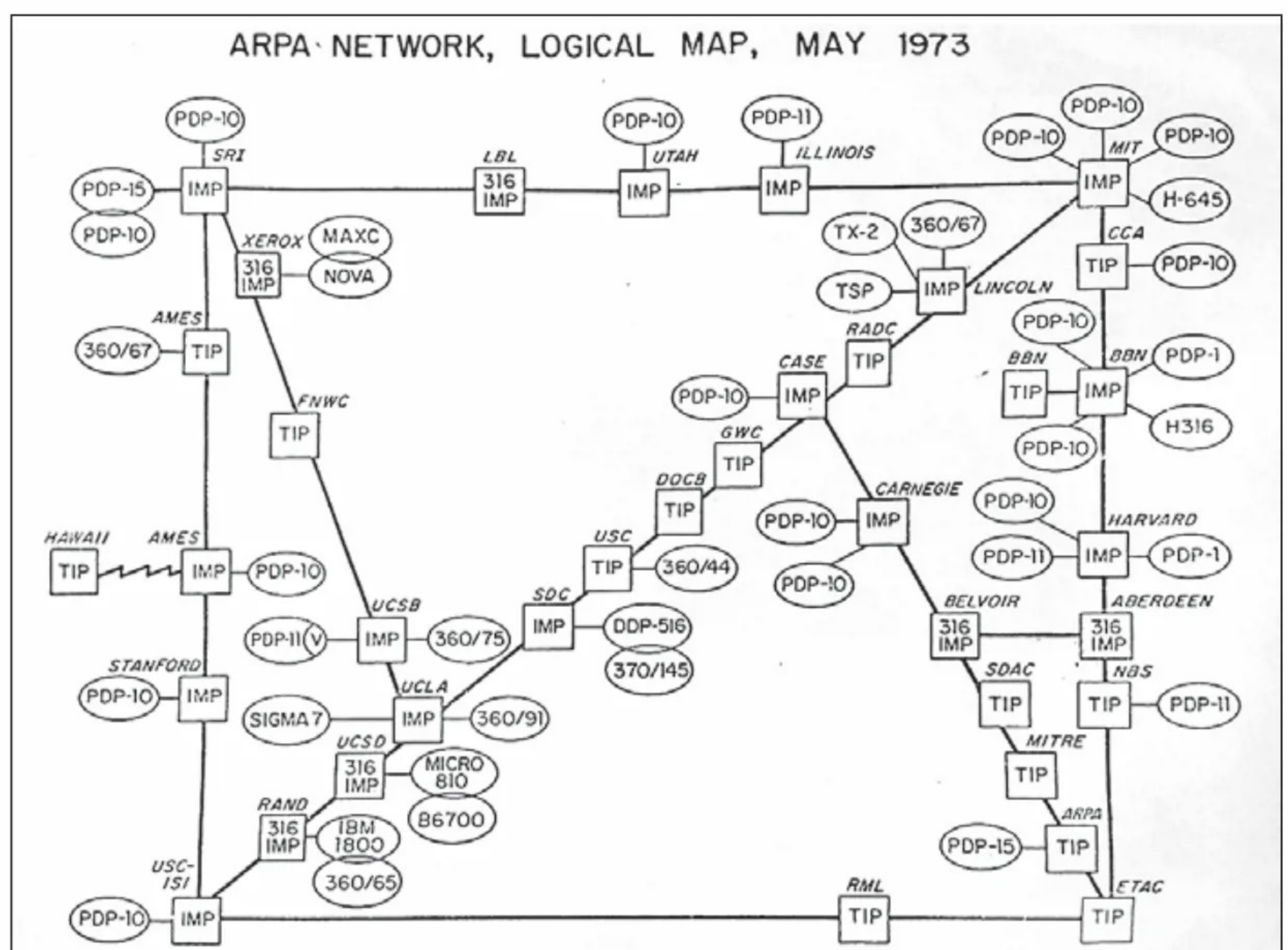


Fig.70: a map of ARPANET, the internet's predecessor, as it appeared in 1973. Source: <https://w.wiki/7FPK>

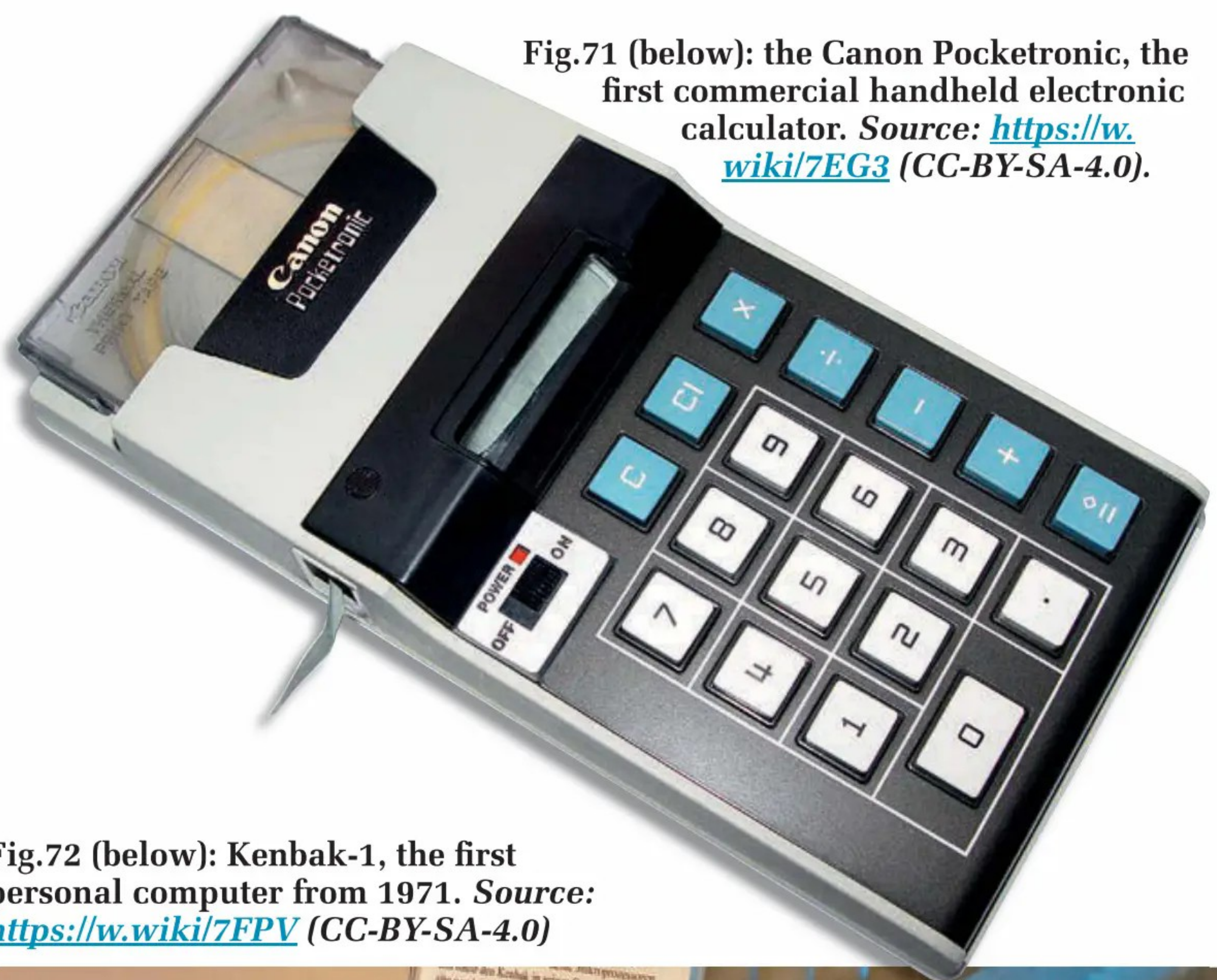


Fig.71 (below): the Canon Pocketronic, the first commercial handheld electronic calculator. Source: <https://w.wiki/7EG3> (CC-BY-SA-4.0).



Fig.72 (below): Kenbak-1, the first personal computer from 1971. Source: <https://w.wiki/7FPV> (CC-BY-SA-4.0)

Feature Article

transmit a page at 96 lines per inch (38 per cm).

Cartrivision, a consumer videotape cartridge format, was introduced. The machines were built into expensive TV sets, which were a commercial failure. See www.angelfire.com/alt/cartrivision/

The Unix operating system was rewritten in the C language, so 1972 could be considered a date when C became mainstream. C was mainly developed between 1969 and 1973 and is still widely used today (in its original form and derivatives like C++ and C#).

The first microprocessor for personal computers was released, the 8-bit Intel 8008.

The world's first scientific pocket calculator was introduced, the HP-35.

Pong, the first commercially successful computer game, was released. We created a modern, miniaturised version that was published as a project in the August 2022 issue of *Practical Electronics*.

The first blue LED was invented at RCA by Herbert Paul Maruska (1944~), but the company was in turmoil and the project was cancelled. Also, the device was too dim for practical use; see <https://pemag.au/link/abp6>

Eventually, Isamu Akasaki (1929-2021), Hiroshi Amano (1960~), and Shūji Nakamura (1954~) won the Nobel Prize in 2014 for their 1993 invention of high-brightness blue LEDs at Nagoya University in Japan.

White LEDs are blue LEDs with

a scintillator coating (similar to a phosphor).

| *SPICE, Ethernet, graphical interfaces* **1973**

Micral released the first personal computer with a microprocessor (the Intel 8008).

The SPICE (Simulation Program with Integrated Circuit Emphasis) analog circuit simulation program was introduced. It and its derivatives (like LTspice) are still widely used today.

Ethernet was invented by Robert Melancton Metcalfe (1946~) and his team working at Xerox Palo Alto Research Center (Xerox PARC) in California. It is one of the key technologies of the internet.

Motorola demonstrated the cellular mobile phone, although it took some time to commercialise.

The first tuneable laser was demonstrated at Bell Labs.

The Xerox Alto computer (Fig.73) was released, the first computer with a graphical user interface and a mouse (see Fig.74), ten years before the Apple Lisa. It cost US\$32,000, equivalent to \$330,000 today. It also had a portrait-orientated display.

| *EDUC-8 computer, CP/M OS etc* **1974**

Electronics Australia published what was thought at the time to be the world's first kit computer, the EDUC-8 (Fig.75), but it was later found to have been beaten by a competitor by one month, the Mark-8. However, the EA design was considered superior.

Bravo was the first 'WYSIWYG' document preparation program, running on the Xero Alto computer, an early word processor.

The CP/M computer operating system was introduced, later displaced by MS-DOS.

| *Kodak digital camera, Betamax etc* **1975**

The first self-contained digital camera was invented by Steven Sasson (1950~) at Kodak. It had a 100×100 pixel resolution and images were recorded digitally on cassette tape, taking 23 seconds.

The Altair 8800 personal computer kit was released, considered by many to have started the microcomputer revolution.

The Betamax home video recording system was released (the SILICON CHIP series mentioned earlier also covers Betamax in detail).

The Steadicam was invented by Garrett Brown and produced by Cinema Products Corporation. It is used for camera stabilisation, as it isolates the

operator's movement from the camera. Silicon Chip described the Steadicam in its November & December 2011 issues (siliconchip.au/Series/33).

| *VHS tape system, 5.25in floppy etc* **1976**

The first word processor for home computers was released, called "Electric Pencil", for use on computers such as Altair 8800, Sol-20 and later, the TRS-80 and the IBM PC.

The VHS home video tape system was released.

5.25-inch (13.3cm) floppy disks became available.

| *Apple II, Commodore PET, TRS-80 etc* **1977**

The first practical optical fibre link was installed in Turin, Italy.

The influential Apple II, Commodore PET and TRS-80 home computers were released.

| *speech synthesis, LaserDisc etc* **1978**

Texas Instruments released the first speech synthesiser chip, the TMS5100. It used "pitch-excited linear predictive coding" to greatly decrease the volume of data required to generate speech. It was used in the "Speak & Spell" educational toy.

The LaserDisc was released on the market. Machines could play prerecorded videos but could not record. Technology from LaserDisc was later incorporated into Compact Discs, DVDs and Blu-rays. It was never hugely popular but offered good-quality video reproduction for the period, far superior to VHS.

| *Fairlight CMI, 1G phone networks etc* **1979**

The Australian Fairlight CMI (Computer Musical Instrument) was released. It was based on a design by Tony Furse, licensed by Kim Ryrie and Peter Vogel (ex ETI magazine). It was "one of the earliest music workstations with an embedded sampler", considered revolutionary at the time. See the video titled "How the Fairlight CMI changed the course of music" at <https://youtu.be/jkiYy0i8FtA>

The very popular WordPerfect word processor was released.

Japan's Nippon Telegraph and Telephone (NTT) deployed the first 1G cellular phone network.

Philips and Grundig released the Video 2000 consumer video cassette format, discontinued in 1988.

The VisiCalc spreadsheet program was released. It was considered a "killer application" for the Apple II and ran on many other computers. It is the predecessor to programs like Excel. For more details, visit: <http://danbricklin.com/visicalc.htm>



Fig.73: the Xerox Alto computer from 1973. Source: <https://w.wiki/7EG4>

The History of Electronics, part 6

Commodore VIC-20 computer etc

1980

The ITU (International Telecommunications Union) Group 3 fax standards (digital) were released. The time to transmit a page was reduced to 6-15 seconds, not including handshaking. It supported a variable scanning resolution, up to 400 lines per inch (157 per cm).

The Commodore VIC-20 computer was released.

MS-DOS V1.0, 16-bit DAC etc

1981

The MS-DOS V1.0 computer operating system was released, along with the IBM PC.

The Osborne 1 was released, it is considered to be the first commercial truly portable/luggable computer. It is not obvious what device should get the credit for the first 'laptop'; many contenders exist.

The PCM53/DAC700 16-bit single-chip audio digital-to-analog converter (DAC) was released. Designed by Jimmy Naylor and a Texas Instruments/Burr-Brown design team, it became the basis of nearly all audio CD players.

RCA released its Capacitance Electronic Disc (CED), an analog video disc playback system. A stylus with mechanical tracking read the disc. The discs were 30cm in diameter and could record 60 minutes of NTSC video per side. The product was unpopular and discontinued due to competition from LaserDisc players and other reasons.

CD player, Commodore 64

1982

The first audio Compact Disc (CD) player (co-developed by Philips and Sony) was released in Japan.

The Commodore 64 computer was introduced.

3.5in floppy disk, C++ language etc

1983

The first 3.5-inch (8.9cm) floppy disks became available, based on the Microfloppy Industry Committee (MIC) specification.

The C++ programming language was released, an 'object-oriented' version of C that's still widely used today.

The first personal computer with a built-in hard disk, the IBM PC XT with 10MB standard capacity, went on the market.

Motorola released the first 'mobile' phone, the DynaTAC 8000X. It weighed nearly a kilogram, took 10 hours to charge and retailed for US\$3995 (about \$18,750 in today's money) – see Fig.76.

Dr Mitsuaki Oshima at Panasonic invented electronic image stabilisation.

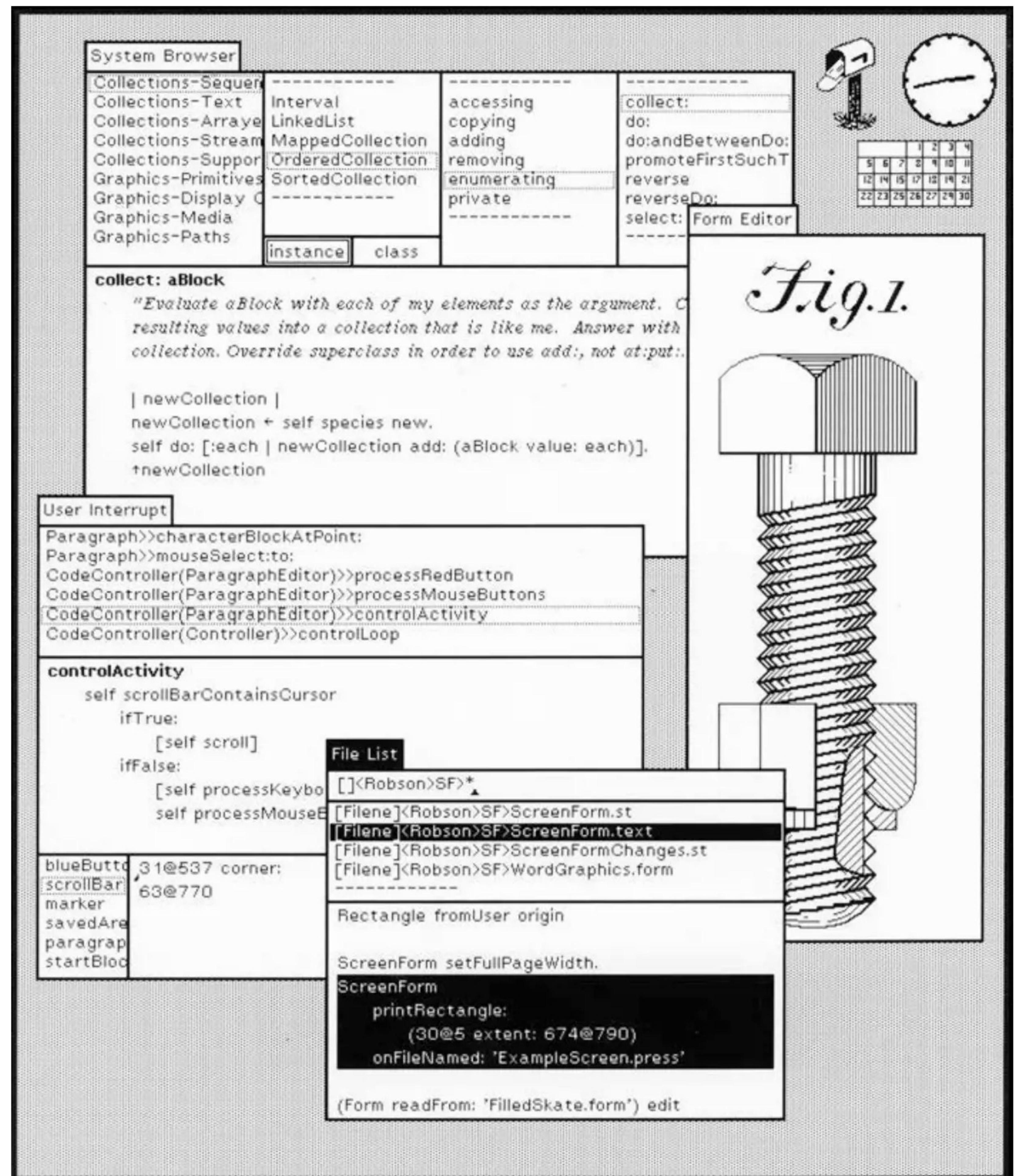


Fig.74: the Xerox Alto GUI from 1973. Source: <https://interface-experience.org/objects/xerox-alto/>

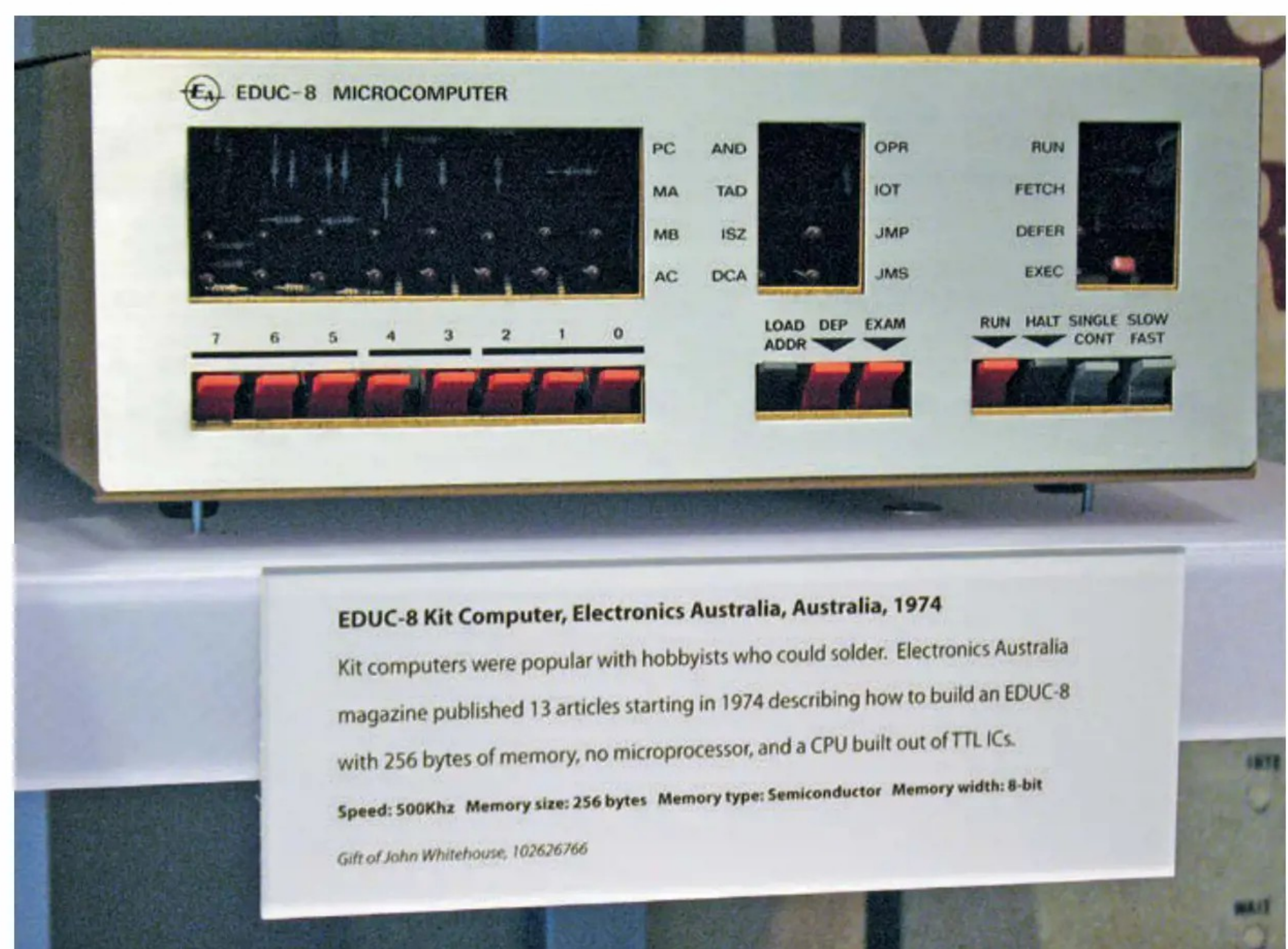


Fig.75: the Electronics Australia EDUC-8 computer. Source: <https://w.wiki/7EG6>

Feature Article



Fig.76: a Motorola DynaTAC 8000X mobile phone. Source: <https://wiki/7FPn> (CC-BY-SA-3.0)

Panasonic released the first video camera to feature electronic image stabilisation later, in 1988.

| Apple Macintosh, CD-ROM

1984

The Apple Macintosh was released. The Commodore Amiga computer was also introduced.

The CD-ROM for data storage, based on the audio CD, was announced.

| IBM Tangora speech recognition

1985

The IBM experimental speech recognition system Tangora became available. It ran on an IBM PC AT and recognised 20,000 words, converting them to text.

| Sony D-1 video recording format

1986

The professional studio Sony D-1 digital video recording format was introduced.

| Higher temperature superconductors

1987

“Higher temperature” superconductors were discovered. Currently, the highest-temperature superconductor works at around -135°C at normal atmospheric pressure.

| TAT-8 transatlantic optical fibre cable

1988

The first transatlantic optical fibre cable, TAT-8, became operational. It had a capacity of 280Mbit/s, equivalent to 40,000 telephone circuits. It was retired in 2002; it rapidly reached capacity when it was ini-

tially thought that its capacity would never be reached. Sharks also attacked the cable, possibly due to them being able to sense its electromagnetic radiation.

It was instrumental in the development of the internet, providing a dedicated high-speed T1 connection between CERN in Europe and Cornell University in the USA.

| GPS receiver, World Wide Web etc

1989

The first commercial handheld GPS receiver was released, the Magellan NAV 1000.

CDMA (Code Division Multiple Access) was demonstrated for cellular telephone systems.

The World Wide Web was invented by Tim Berners-Lee and released to the public in 1991.

| DragonDictate speech recognition

1990

The first consumer speech recognition software, DragonDictate, was released. Nowadays it's called “Dragon NaturallySpeaking” and is now owned by Microsoft.

| 2G networks, Linux, Python

1991

2G (GSM) telephone networks were introduced.

The Linux operating system for computers, a free/open-source version of Unix, was released.

The Python programming language was released.

| TASMAN2 cable, Windows 3.1 etc

1992

Australia's first undersea optical fibre, TASMAN2, connected us to New Zealand with a speed of around 1Gbps.

Windows 3.1 was released, marking a shift away from the command-line DOS interface on PCs towards graphical interfaces.

The Apple Newton MessagePad was released, an early ‘personal digital assistant’ with handwriting recognition that helped form the basis of later smart devices.

| Windows NT, HAARP

1993

Windows NT was released. Its core still underlies modern Windows versions such as 10 & 11. However, its GUI was still similar to that of Windows 3.1.

HAARP (High-frequency Active Auroral Research Program) was established for upper atmosphere and ionospheric research.

| CompactFlash memory cards etc

1994

The first CompactFlash memory cards were produced by SanDisk, starting at 2MB. It was the first widespread, dedicated flash memory card format.

Apple released home and office

computers using IBM's 32-bit PowerPC processors, marking a shift away from the Motorola processors they previously used.

IBM released the Simon Personal Computer (SPC), the first ‘smartphone’, although that term didn't exist at the time.

It had an LCD touchscreen and could be used to make or receive phone calls, send and receive faxes, emails and pages (‘instant messages’). 50,000 were sold for US\$1099 (about \$3500 today).

| Windows 95, DAB radio

1995

Windows 95 was released, with a GUI reminiscent of modern Windows versions.

DAB digital radio broadcasting began in Europe.

| DVD player, PalmPilot “smartphone”

1996

The first digital video disc (DVD) player was released in Japan.

The ATSC digital television standard was released.

The PalmPilot was released, an early predecessor to the modern smartphone.

| MPMAN F10 portable MP3 player

1997

The DVB-T digital television standard was released, with the first broadcast in Sweden.

The first portable MP3 player was released, the MPMAN F10 by Saehan Information Systems.

| ADSL standard

1998

ADSL (Asymmetric digital subscriber line) technical standard ANSI T1.413 Issue 2 was released. ADSL enabled high-speed data over standard copper telephone lines. It was introduced in Australia in 2000.

| Bluetooth devices

1999

The first Bluetooth device was introduced to the market.

| SD memory cards, Windows 2000

2000

The first SD (Secure Digital) memory cards were released with 32MB and 64MB capacities.

Windows 2000 was released, merging the core of Windows NT with the graphical interface of Windows 95. It was the basis of the modern Windows operating system in 2023.

| 3G networks, Mac OS X, iPod

2001

3G telephone networks were introduced, offering high-speed mobile data, up to 7.2Mbps.

Apple released Mac OS X, a clean-sheet redesign of their graphical operating system based on FreeBSD, still their primary operating system today.

Apple released the iPod MP3 player.

The History of Electronics, part 6



- Project boxes designed and manufactured in the UK.
- Many of our enclosures used on former Maplin projects.
- Unique designs and sizes, including square, long and deep variations of our screwed lid enclosures.
- Sub-miniature sizes down to 23mm x 16mm, ideal for IoT devices.



www.terrington-components.co.uk | sales@terrington-components.co.uk | Tel: 01553 636999

ISDB-T digital TV broadcasts

2003

Japan started digital TV broadcasts using the ISDB-T standard.

LongPen remote signing device

2004

Margaret Atwood (1939~) invented the LongPen, a remote signing device, mainly for authors to sign copies of books. It was released to the market in 2006. It is reminiscent of Elisha Gray's telautograph from 1888.

DMB standard

2005

South Korea adopted DMB (Digital Multimedia Broadcasting), for mobile video streaming, a development of the DAB radio broadcasting standard.

DTMB standard, OPAL reactor

2006

The DTMB (Digital Terrestrial Multimedia Broadcast) TV standard was adopted in China.

OPAL (the Open-pool Australian lightwater reactor) was commissioned to replace HIFAR for research and radioisotope production (eg, for medical procedures and industrial applications).

Apple iPhone

2007

Apple introduced the iPhone, the first truly modern smartphone (the Blackberry was released in 2000, but phones with inbuilt keyboards eventually fell out of favour).

Further reading

- "Phonogram Images on Paper, 1250-1950" at <https://youtu.be/TESkh3hX5oM>
- "Experiments and Observations on Electricity" made at Philadelphia in America by Benjamin Franklin, 1751 – <https://pemag.au/link/abpf>
- Simple construction project video: "Voltaic Pile, the First Battery" at <https://youtu.be/pW4UU0gJX6k>
- "Electric Incandescent Lighting" by Edwin James Houston and Arthur Edwin Kennelly, 1896 – <https://pemag.au/link/abpe>
- "The Progress of Invention in the Nineteenth Century" by Edward W. Byrn, 1900 – <https://pemag.au/link/abpc>
- "A History of Wireless Telegraphy" by J.J. Fahie, Third Edition, 1902 – <https://pemag.au/link/abpd>
- "How does a spark gap transmitter sound?" at <https://youtu.be/VMdYte66D2Y>
- The First Digital Voltmeters and the Birth of Test Automation – www.hp9825.com/html/dvms.html
- The oldest surviving video recording: "The Edsel Show - CBS-TV (October 13, 1957)" at <https://youtu.be/Ze0Az9tdkHg>
- "Oldest surviving color videotape recording: WRC-TV dedication May 22, 1958" – <https://youtu.be/4vBEMGTdDYc>

DAB+ broadcasting began, starting in Australia, using less bandwidth for similar audio quality to DAB.

4G networks

2009

4G (LTE) telephone networks were introduced.

Apple iPad

2010

Apple released the iPad, an early touchscreen tablet computer.

5G networks

2019

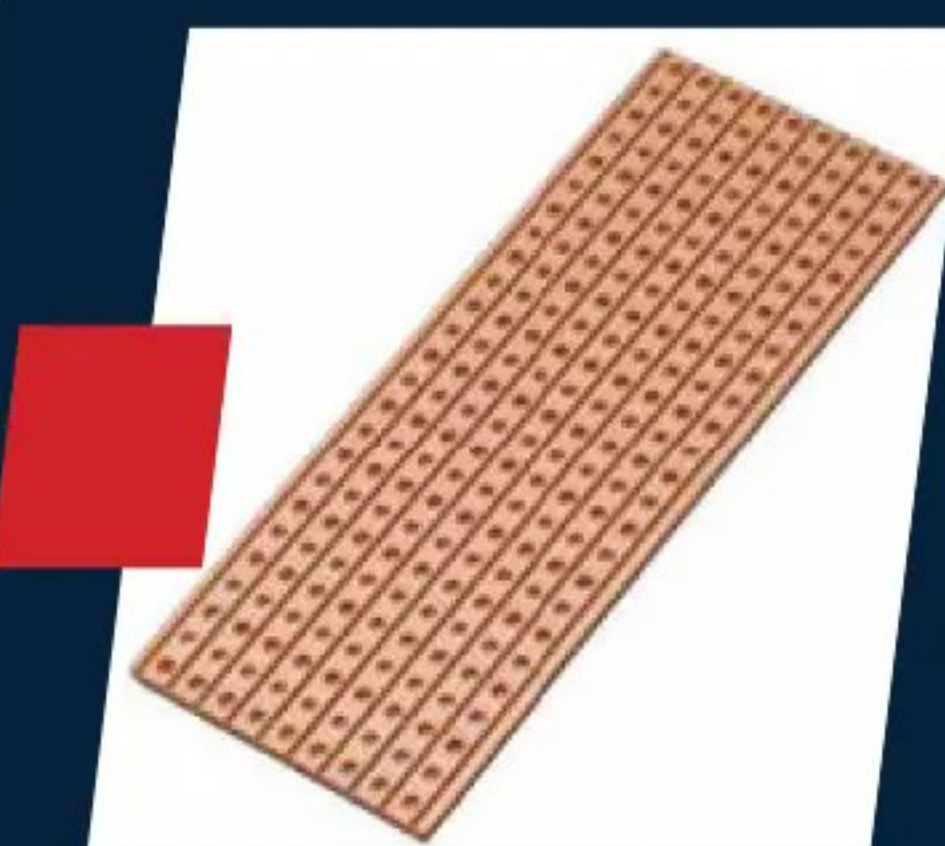
5G telephone networks were introduced.

Apple Silicon (ARM CPU)

2020

Apple brought to market computers using its own Apple Silicon processors, the M1 series, using memory package stacking for high performance with low power consumption. **PE**

FIND ALL YOUR ELECTRONIC COMPONENTS IN ONE PLACE



01623 720730

Available from 9AM - 5PM Mon-Fri



BASIC MICRO



ELECTRONICS COMPONENT SUPPLIER

WWW.BASICMICRO.CO.UK

High-quality, genuine parts

INTELLIGENT DUAL HYBRID POWER SUPPLY

PART 1: BY PHIL PROSSER

This power supply has two separate outputs, each capable of delivering up to 25V DC at 5A. They can be connected in series and ganged up to form a dual tracking supply, and both outputs are controlled and monitored using a graphical LCD screen, two rotary encoder knobs and two pushbuttons.

Both outputs are powered by a single transformer, and they can be used independently or ganged up to form a dual-tracking (positive and negative) or higher current single-ended supply.

This design uses a hybrid switch-mode/linear approach for decent efficiency and low output ripple and noise. Due to its high efficiency, it doesn't need fans, so there is no fan noise or associated dust buildup.

Much audio and analog work demands a bench power supply with decent voltage and current capability, plus dual tracking outputs, so this supply fits the bill.

We received some questions on the practicality of building a pair of our 45V, 8A linear supplies (October & November 2020) and hooking them together to make a dual supply.

You certainly could do that, but this supply is a much more compact

and lower cost solution. It adds valuable features like monitoring the voltages and currents on one screen, and switching off or reducing the voltage of both outputs if either current limit is exceeded.

The slightly lower voltage and current capabilities (25V instead of 45V and 5A instead of 8A) will still suit most applications. For example, while this supply won't allow you to test a 100W power amplifier module at full power, it would be good enough to test it at lower power levels, to verify that it works before hooking up its normal power supply.

And when you aren't using it as a tracking supply, you can make the two outputs completely independent and control them separately.

Another advantage of the digital controls is that the internal wiring for this supply is quite straightforward and neat, consisting mainly of

some ribbon cables that carry control signals, plus a handful of wires that carry DC power.

Using a microcontroller to control the power supply and drive the user interface allows us to be smart in how we control the limits. It can work out voltage and current limits based on the transformer's VA rating and secondary voltage. This allows a wide variety of transformers to be used. Dig through your parts bin and recycle!

The supply uses two alike regulator boards for dual rails. It can be built with a single board if you only need one rail – the user interface can handle single-/dual-rail implementations.

If you're dead set against using a microcontroller, the regulator board has been designed so that it can operate with just two pots. You would need to organise your own voltage and current monitoring, but you can build it that way, and leave out quite a few

Intelligent Dual Hybrid Supply part 1

of the more expensive parts, like the analog/digital conversion chips, isolators, CPU and display.

The microcontroller interface is simple to use, though. There are just two controls you will use day-to-day: the output voltage and current limit. If you need it, there is more detail accessible in setup menus, including calibration and configuration screens.

The interface is controlled using two rotary encoders with integrated push-buttons, plus two extra pushbuttons. The encoders adjust the voltage and current limits, while pressing either swaps between controlling the two outputs.

One of the extra switches lets you go into setup mode, while the second button is an 'emergency stop' button that shuts down the power supply output immediately. This is useful if the magic smoke starts leaking from something! Pressing it again restores the output.

Performance

When measured using an oscilloscope, mains-related hum and buzz is not detectable (see Fig.1), nor is switchmode noise. Output noise is typically less than 20mV peak-to-peak, and less than 5mV RMS. This is pretty much constant across the full range of load variations.

The response of the power supply to load change is good. Figs.2 & 3 show that the output voltage recovers within 100µs with a 5A load step, with a maximum offset of just 200mV over 40µs.

Fig.4 shows how the unit behaves when it goes into and out of current limiting, with the current limit set to 5A. In response to a short circuit on the output, the voltage falls to achieve the programmed current limit almost immediately, and remains stable. Recovery takes around 5-10ms and has very little overshoot.

The supply has no thermal problems when short circuited. With both channels delivering 5A continuous into a short circuit, the heatsink will get quite hot to touch, but settles at about 60°C.

Hybrid design

This supply uses both switchmode and linear regulators. A few quick sums show that a purely linear power supply delivering $\pm 25\text{V}$ and 5A would

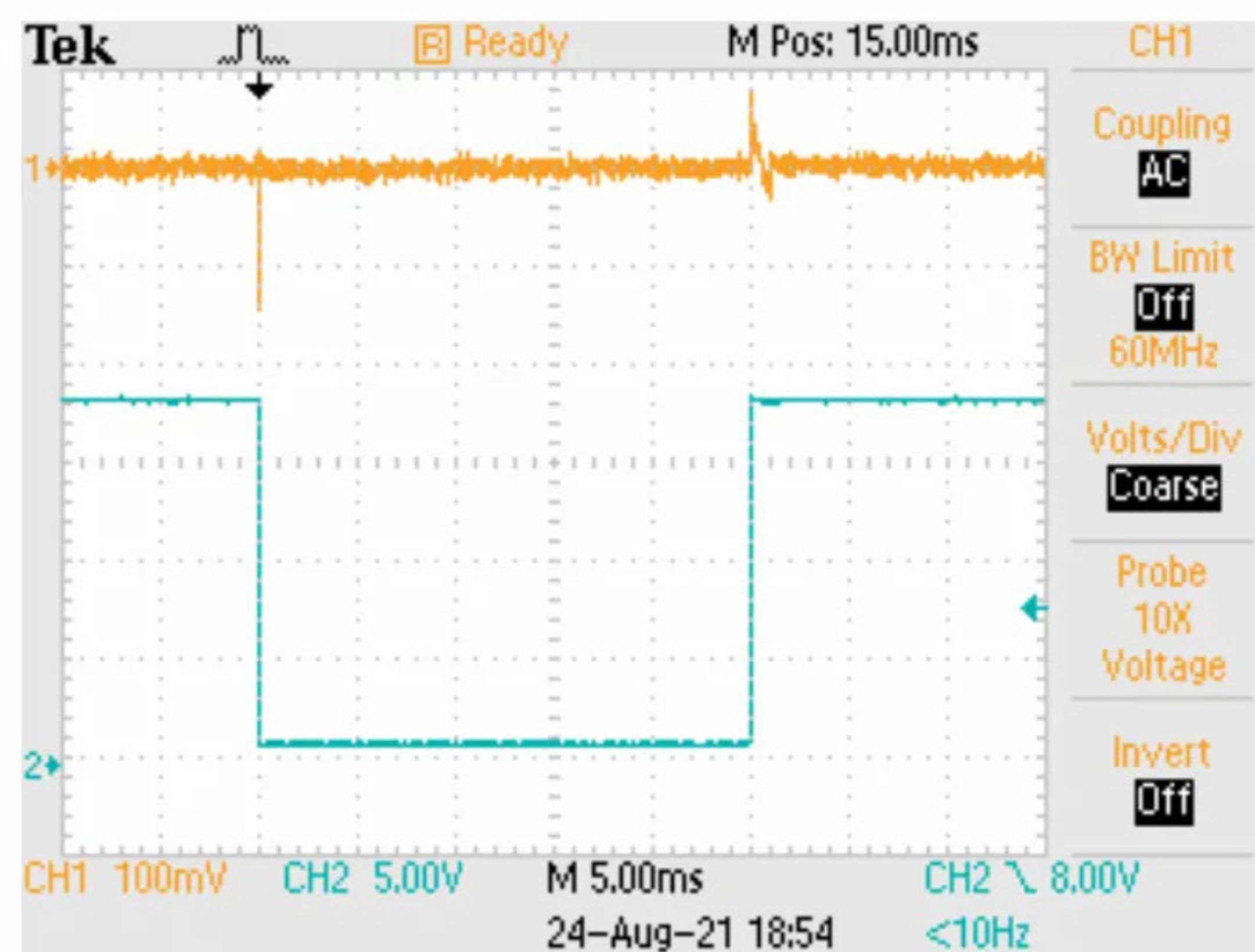


Fig.1: the blue trace shows a 2A load step with the supply set to deliver 15V. The yellow trace is a close-up of the output voltage, showing how it varies. The vertical scale is 50mV/div, and the output voltage only varies by a small amount when the load changes.

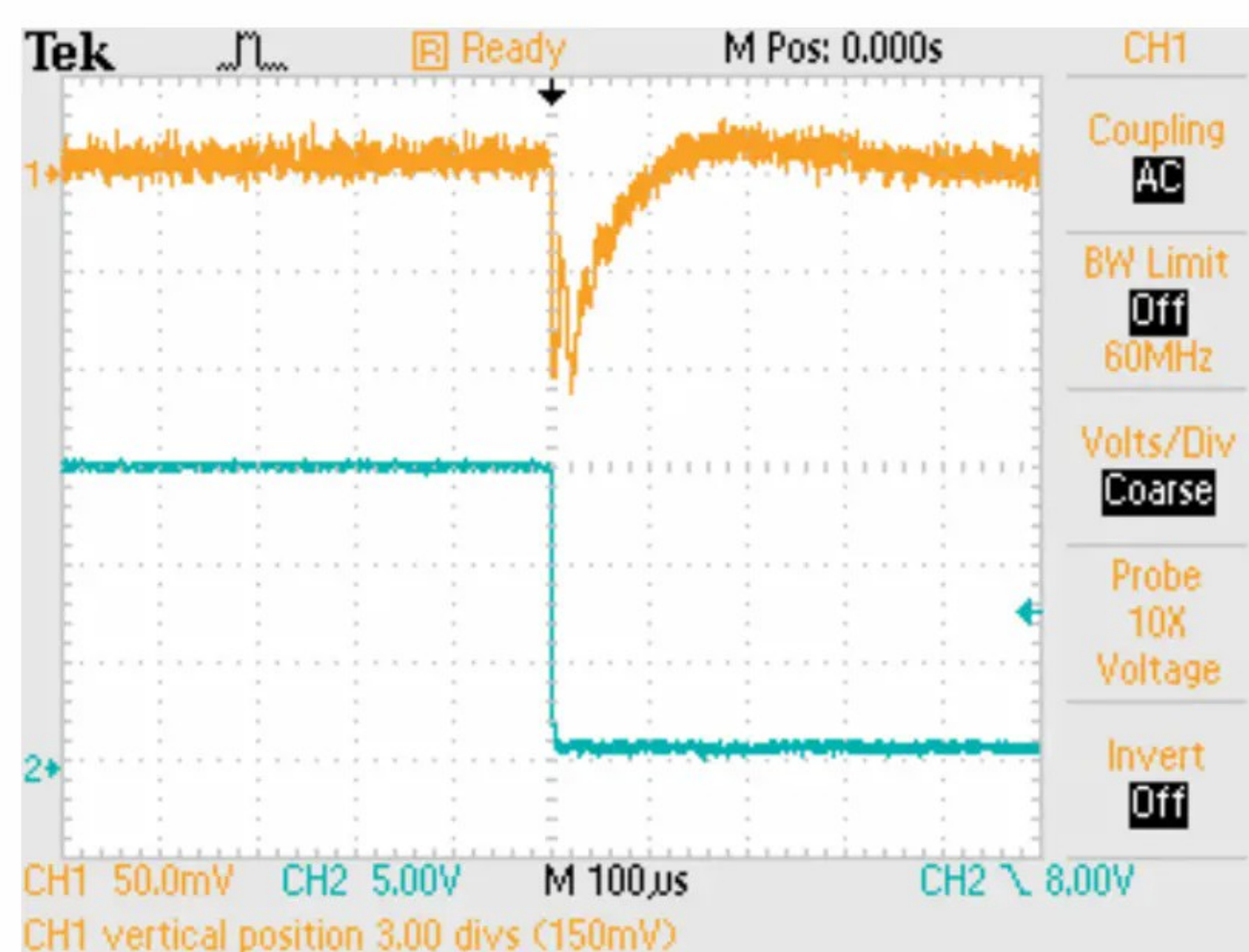


Fig.2: this is a similar view to Fig.1 but with a much faster timebase (100µs per division). The initial 100mV step is characteristic of the LM1084. The LM1084 and the overall loop feedback response brings the output back to 15V within 100µs.

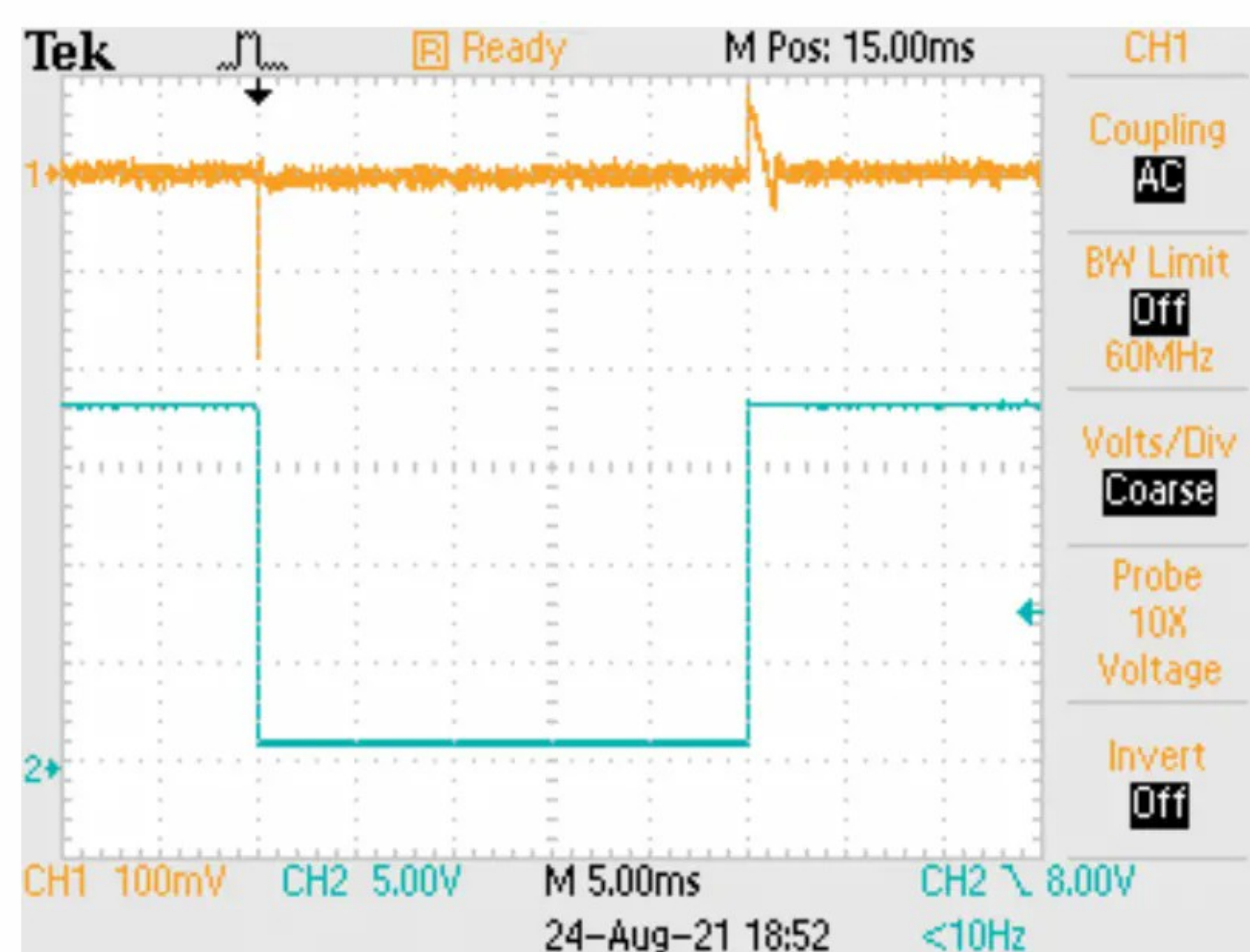


Fig.3: the same scenario as in Fig. 2 except this time, the output voltage has been set to 18V and the load step is 4A. The change in output voltage is slightly greater at 200mV peak drop, recovering within 100µs. On the trailing edge, the output changes by 75mV and it recovers within 2ms. This peak is small for such a large load step with minimal output capacitance.

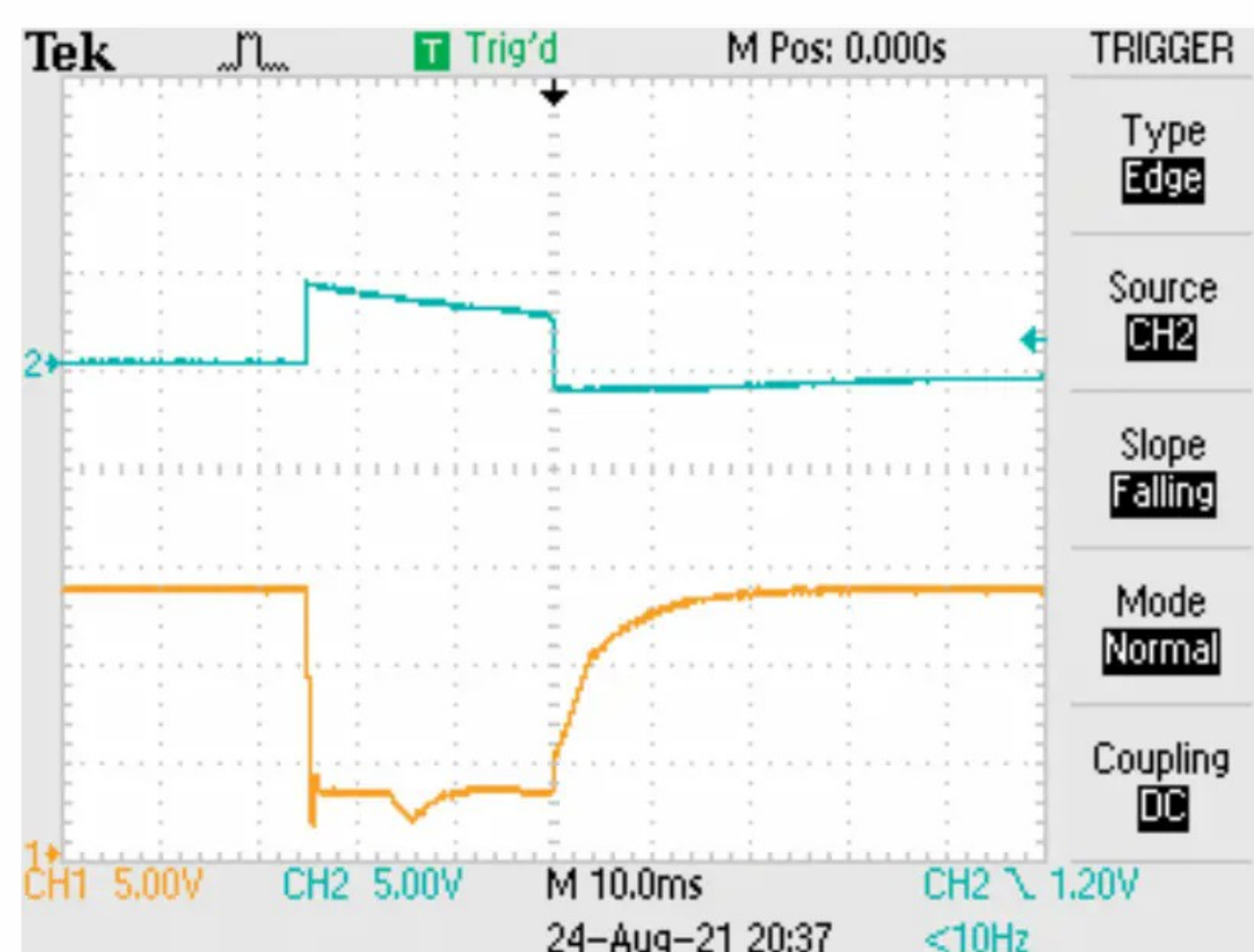


Fig.4: this shows how the unit behaves going into and out of current limiting. Ideally, its reaction should be swift and with little overshoot. In response to a short circuit, the output voltage is rapidly reduced. When the short is removed, the output voltage recovers in about 20ms, with no overshoot visible.

Constructional Project

demand a huge heatsink, dissipating over 125W per rail or 250W total. This is greatly reduced by using a switchmode pre-regulator, which generates just a little more voltage than the linear regulator needs at its input. We aimed for about 5V of headroom in this design.

If we can achieve this, then the linear regulator dissipation is a maximum of $5V \times 5A = 25W$ for regular operation per rail, totalling 50W in the worst case. That is still a reasonable amount of heat to dissipate, but eminently doable. The pre-regulator and bridge rectifier dissipate some power too, which will add in the region of 10W.

The downside is that switchmode power supplies have a reputation of being hard to design, and because of how they work, a bad rap for introducing noise into circuits. Our goal was a product that could be built from standard components, which would 'just work'.

We tried and rejected two alternative pre-regulator designs before settling on the one presented here.

The result meets the above design brief, and neatly fits two independent regulators in the same case. It can deliver 5A over the range of 2-25V continuously per rail, without the need for fans and cutouts.

Implementation

The Intelligent Power Supply comprises four main parts: the main transformer, one or two regulator modules and a controller, as shown in Fig.5. This allows either single or dual rail power supplies to be built.

We expect that most constructors will build the power supply as a dual unit. Each regulator module can operate independently, and its outputs are floating with respect to the other. So for a dual-tracking power supply, you connect the "+" of the negative rail to the "-" of the positive rail and select "Dual Tracking" in the setup.

You can also set the mode to "independent" in the user interface, and independently set voltage and current limits for each rail.

To keep construction simple, we have built a +5V DC power supply for the control interface into the regulator modules. So, the control microcontroller can be powered without the need for separate boards or transformers. Only one of these needs to be installed and enabled.

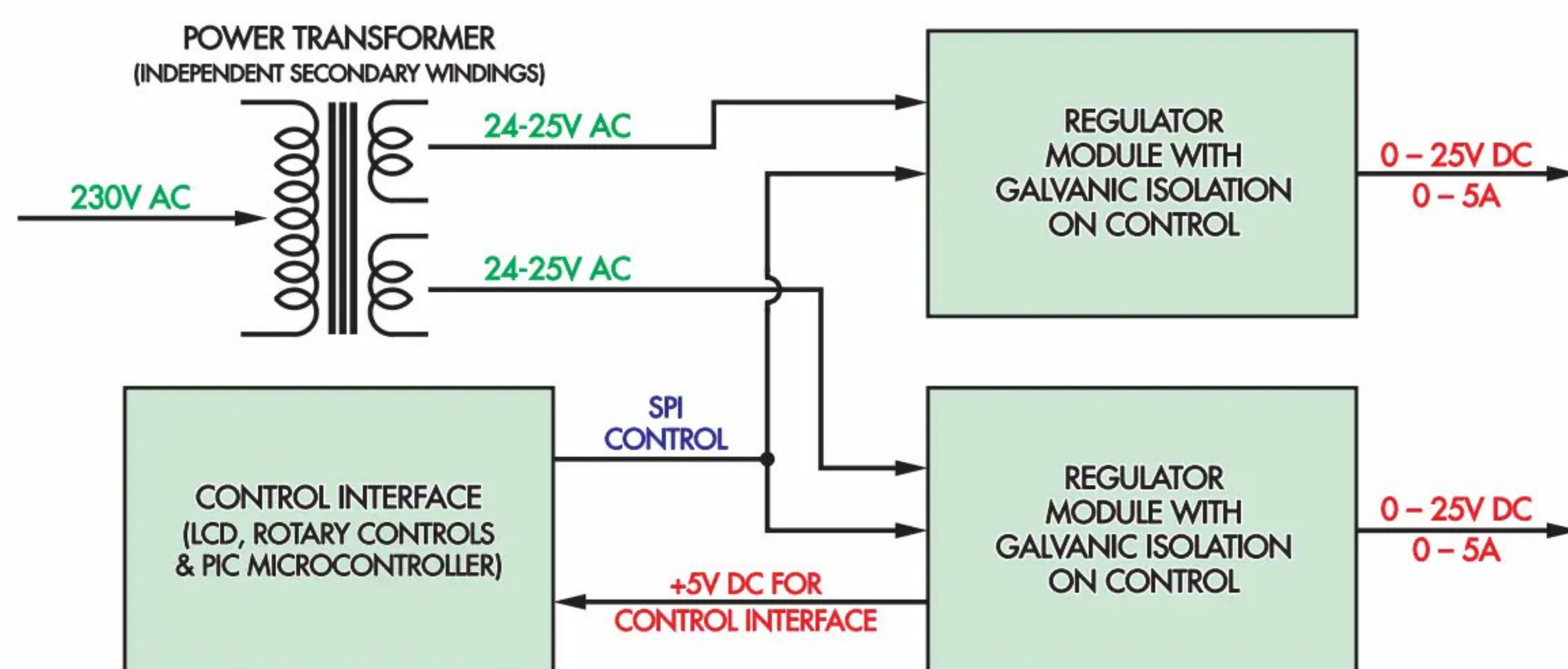


Fig.5: the basic arrangement of the Intelligent PSU. Two separate secondaries on the transformer power the two regulator modules. One of these also provides 5V to the control interface, which uses a serial peripheral interface (SPI) bus to control and monitor both regulator boards.

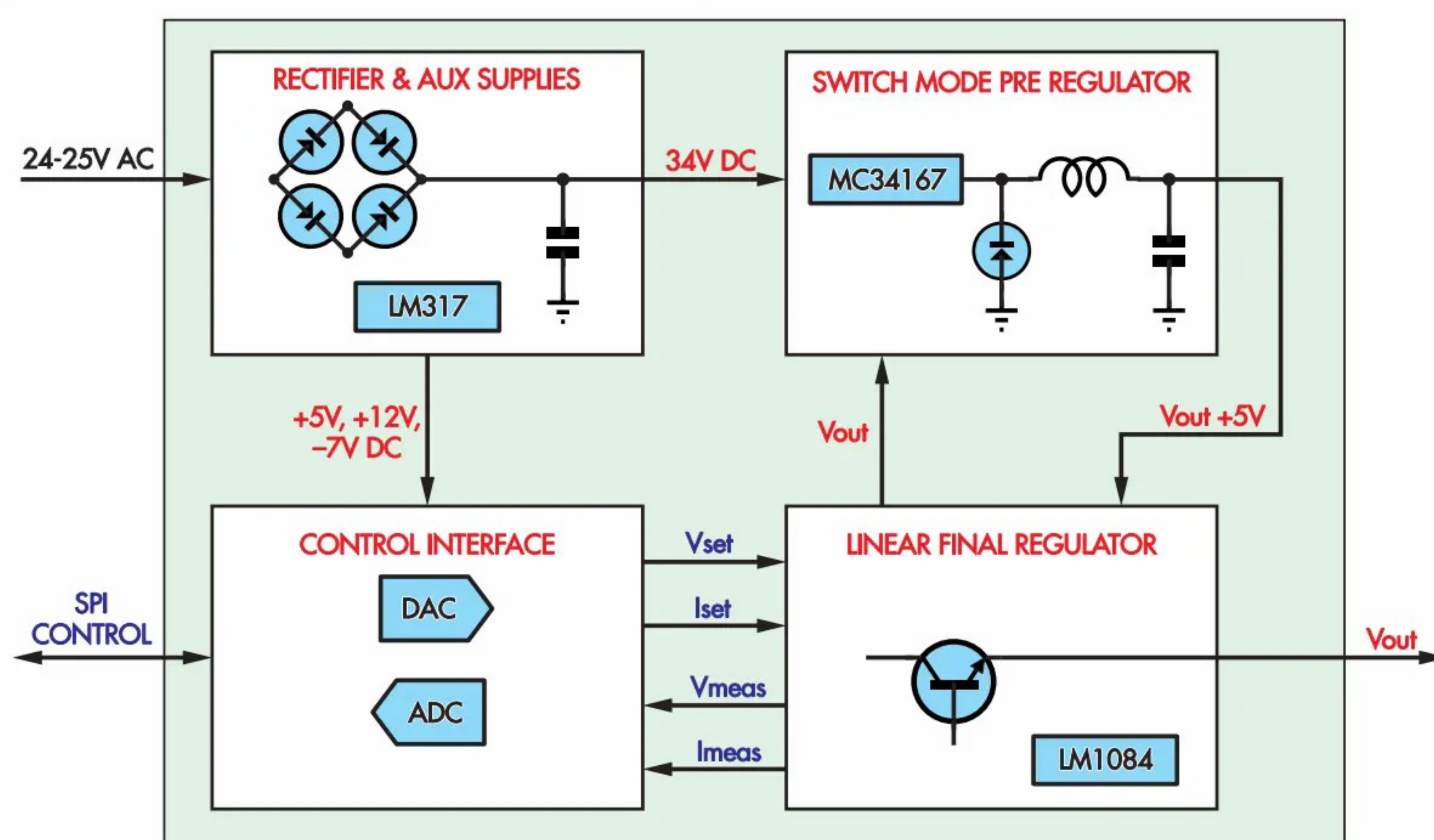


Fig.6: here is how each regulator module is arranged. The incoming AC is rectified, filtered and regulated to provide three supply rails for the rest of the circuitry on the regulator board. The raw DC is also fed to a switchmode pre-regulator which provides 5V more than the selected output voltage to the LM1084-based final linear regulator stage. The output voltage and current are set by a dual-channel DAC, and monitored via a dual-channel ADC.

Refer to Fig.6, the functional block diagram of the regulator module. The regulator takes a nominal 24-25V AC input and control input, and produces regulated DC as commanded.

Our software controls one or two of the regulator modules via a single 10-pin header on each. You could theoretically build more than two, provided you modified our code or wrote your own user interface. We'll explain how to do that later.

As shown in the photos, the module's size (built on a 116 x 133mm PCB) is quite modest for a power supply of this sort. Two of these modules fit side-by-side in the proposed case.

The main heatsink runs across the back of the regulator module(s). Attached to it are two linear regulators, the bridge rectifier and switchmode pre-regulator.

Circuit description

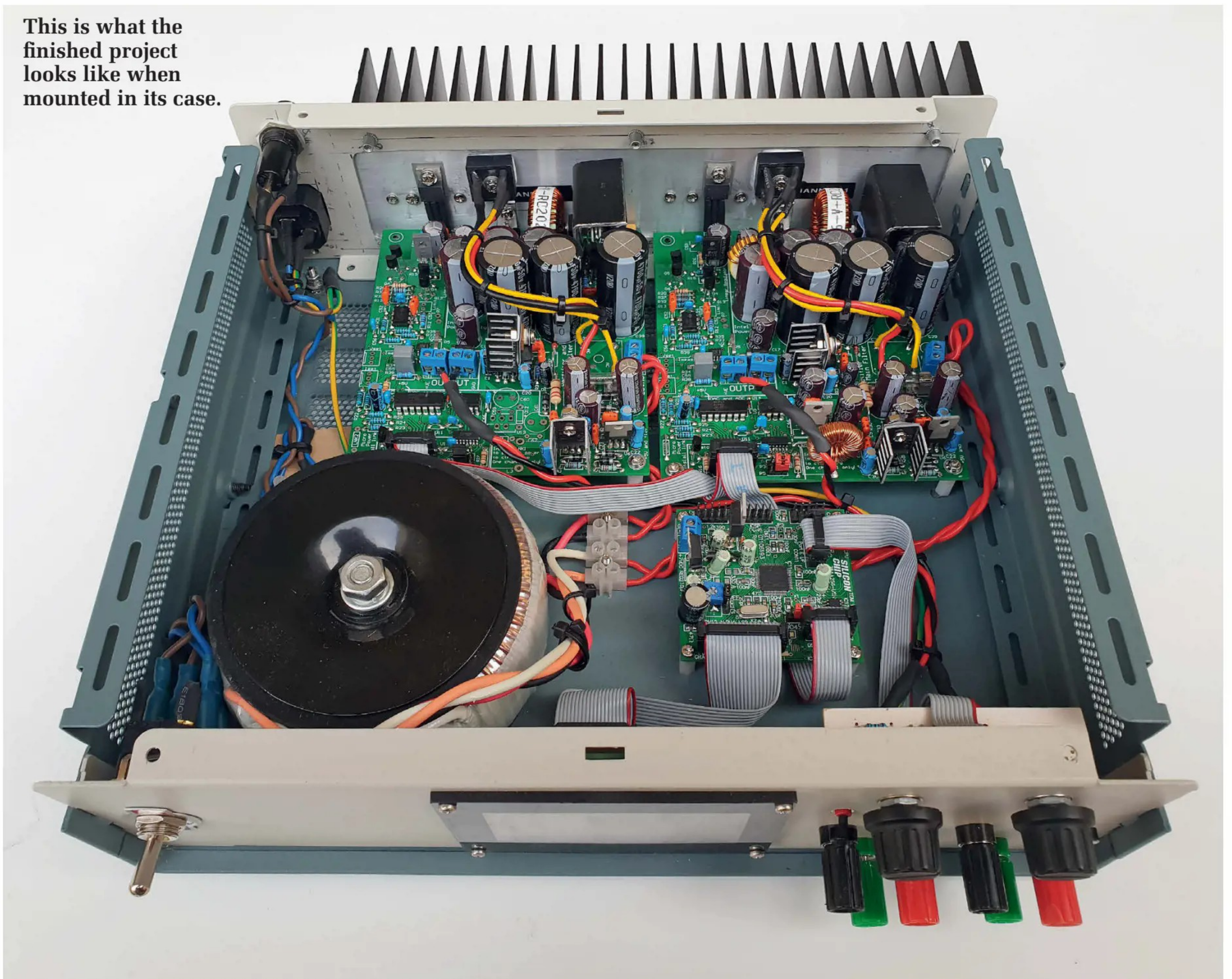
Let's start at the output and work backwards. The complete circuit of one regulator module is shown in Fig.7, and the output regulators are just to the right of the diagram's centre.

The output stage is based on one or two LM1084IT-3.3 regulators. This is a 3.3V low-dropout linear regulator in a TO-220 package. At 5A load, it has a dropout of 1.5V. This low dropout voltage is required to allow the small pre-regulation difference, and get 25V DC from this unit when using a 24-25V AC transformer.

The LM1084IT-3.3 linear regulator from Texas Instruments handles a maximum input-output voltage differential of 25V, although, in this application, the differential will typically be about 5V. The exception is when the current limit kicks in, and while

Intelligent Dual Hybrid Supply part 1

This is what the finished project looks like when mounted in its case.



the pre-regulator capacitors discharge, the LM1084 will see an increased input voltage.

We have specified two LM1084IT-3.3 devices in parallel, with 0.05Ω current-sharing resistors, to ensure that there are no limitations on the output current and to optimise the thermal design.

The output voltage is set with the help of LM358 op amps IC3a & IC3b. IC3b monitors the output voltage, divided by the $15k\Omega$ and $1k\Omega$ resistors, and compares this to the voltage from pin 14 of IC4, a digital-to-analog converter (DAC), labelled V_{SET} .

If the output falls below V_{SET} , it turns off NPN transistor Q6, which allows the voltage at the “GND” pin of the LM1084s (not connected to GND...) to increase. The opposite occurs if the output voltage is too high.

This operational amplifier operates as an integrator, reacting slowly to establish the overall output volt-

age. The high-speed aspect of regulation is dealt with by the LM1084 regulators.

Current control is implemented in the same manner, but instead of monitoring the output voltage, we monitor the output of the INA282 current sense amplifier and compare this to the I_{SET} DAC output (from pin 10 of IC4). If the measured current exceeds the set current limit, NPN transistor Q5 is switched on, pulling the “GND” pin of the LM1084s down.

How do we achieve a 0V output given the minimum voltage an LM1084IT-3.3 can output is 3.3V? This design connects the op amp negative rail and emitters of transistors Q5 & Q6 to a -4.5V rail, allowing the GND pins of the LM1084IT-3.3s to be pulled negative. As a result, the output voltage goes down to 0V.

This part of the circuit is very similar to that published in the 45V Linear Bench Supply project from October

2020 (p20). As in the original article, we have a constant current source comprising two NPN transistors to ensure a minimum load on the LM1084s.

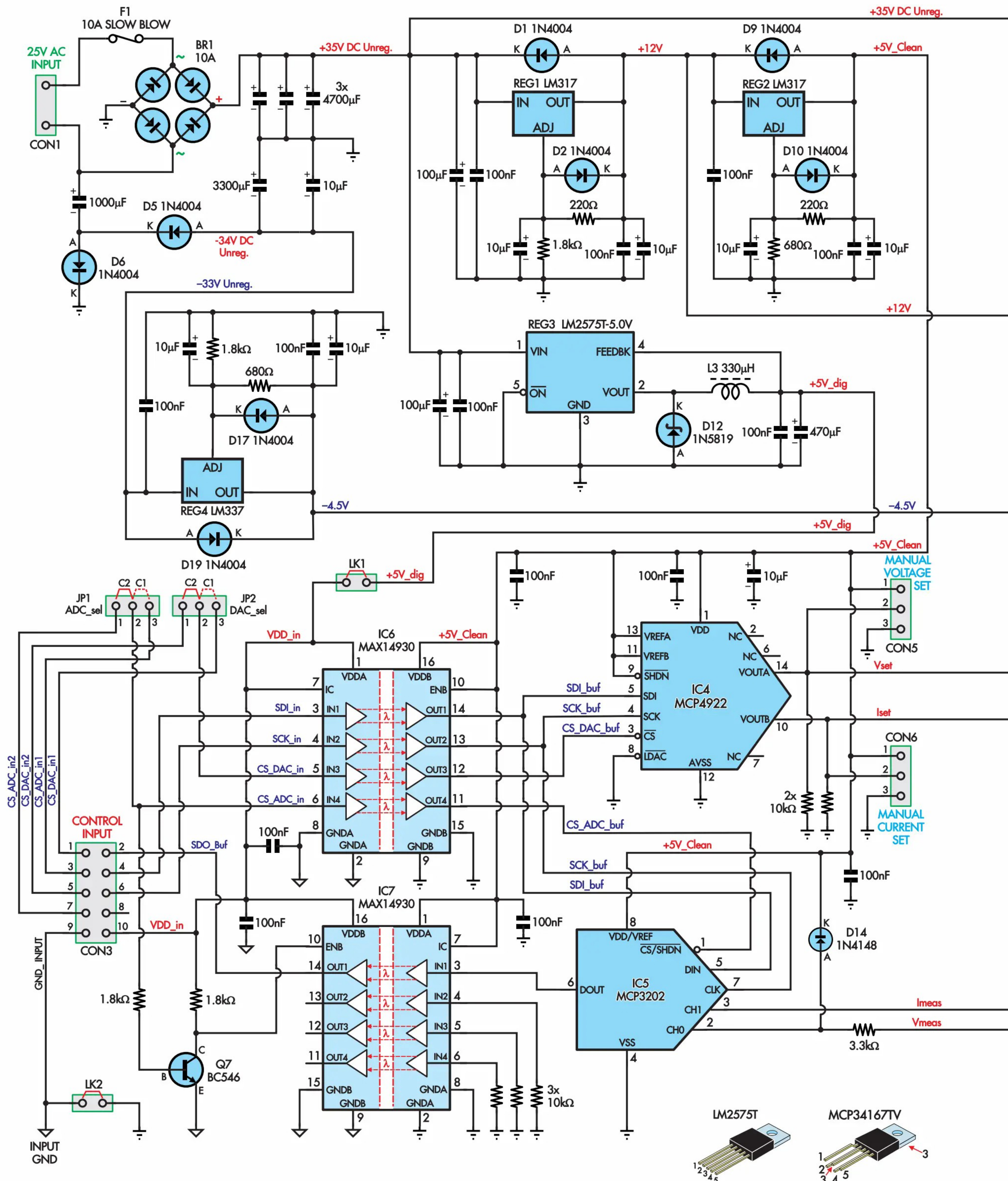
The pre-regulator

We have selected the MC34167 chip as the pre-regulator. This is a switchmode ‘buck regulator’ (step-down) which operates at about 72kHz. A buck regulator switches the input voltage (pin 4) through to the output inductor (pin 2) on and off rapidly. There are two distinct phases of operation in a buck regulator:

When the regulator switch is on, current flows from the input rail (34V DC), building up the inductor current and charging the output capacitor. The inductor stores energy in its magnetic field as a function of the current passing through it.

When the regulator switch is off, current continues to flow through the inductor, as is required because there

Constructional Project



DUAL HYBRID POWER SUPPLY

REGULATOR CIRCUIT

Intelligent Dual Hybrid Supply part 1

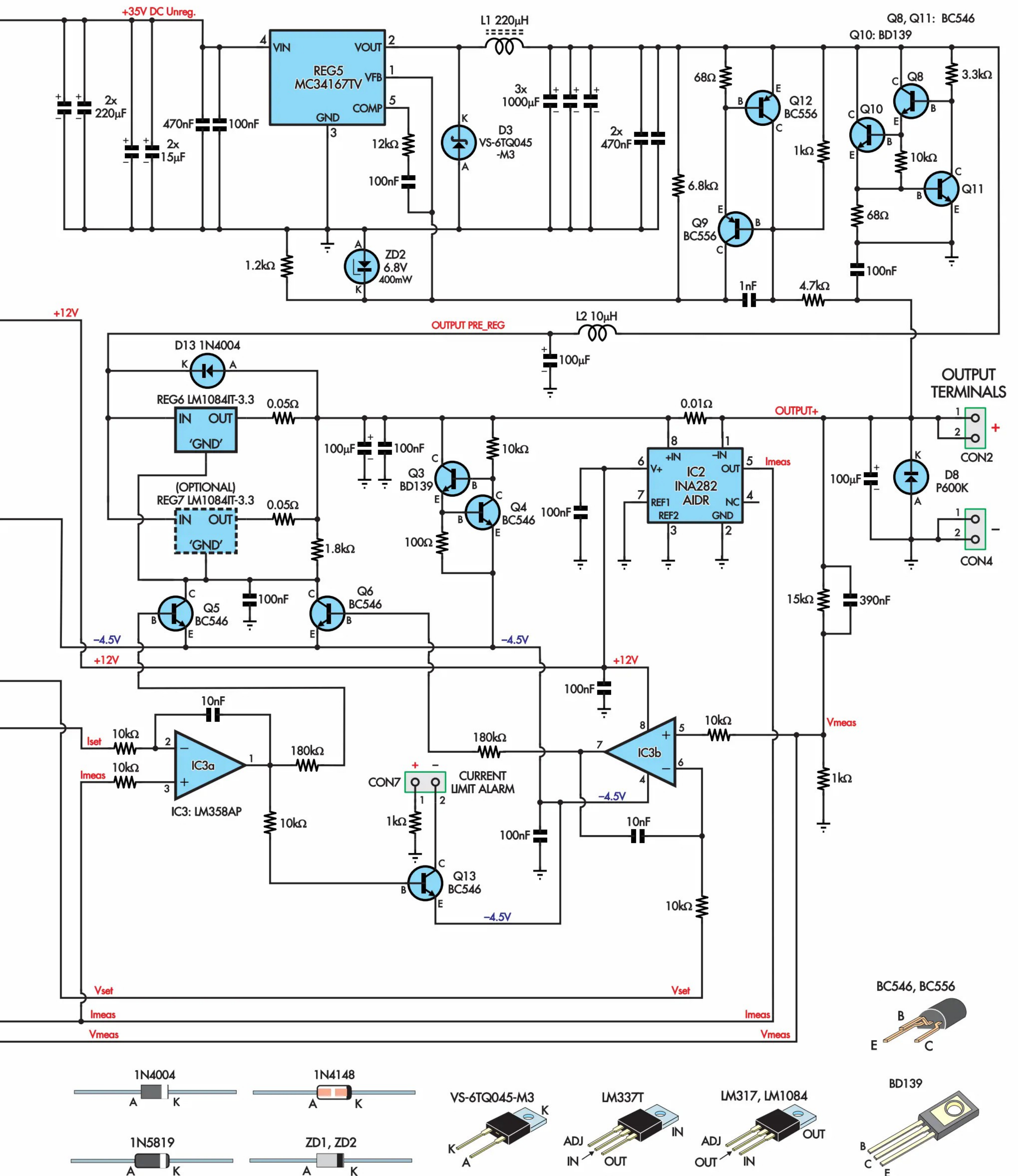


Fig.7: this shows the entire regulator module circuit. The rectifier, filter and regulators that provide the +12V, +5V & -4.5V rails are at upper left. The ADC, DAC, and isolating circuitry are at lower left. The switchmode pre-regulator is at upper right, and the final linear regulator stage and current monitoring circuitry are at middle/lower right.

Constructional Project

Parts List – Dual Hybrid Power Supply

1 metal instrument case, minimum 305 x 280 x 88mm
1 CPU board assembly (see below)
1 LCD assembly (see below)
1 front panel interface assembly (see below)
2 regulator assemblies (see below)
1 230V AC to 24-0-24 or 25-0-25 160-300VA toroidal transformer (T1) [eg, Farnell [1675089](#)]
1 chassis-mount 10-13A IEC mains input socket
1 10A-rated safety 3AG panel-mount fuseholder
1 10A fast-blow 3AG fuse
1 300 x 75 x 46mm diecast aluminium heatsink
24 M3 x 16mm panhead machine screws
16 M3 x 6mm panhead machine screws
14 M3 hex nuts
12 flat washers, ~3.2mm ID (to suit M3 screws)
22 shakeproof washers, ~3.2mm ID (to suit M3 screws)
12 fibre or Nylon washers, ~3.2mm ID (to suit M3 screws)
3 ~3.2mm inner diameter solder lugs (to suit M3 screws)
2 20-way IDC line sockets
5 10-way IDC line sockets
1 4-way 17.5A mains-rated terminal block
2 100nF 63V MKT capacitors
2 10nF 63V MKT capacitors

Wire, cable etc

1 2m length of red 7.5A hookup wire
1 1m length of black 7.5A hookup wire
1 1m length of yellow 7.5A mains-rated hookup wire
1 1m length of green/yellow striped 7.5A mains-rated hookup wire ★
1 1m length of brown 7.5A mains-rated hookup wire ★
1 1m length of light blue 7.5A mains-rated hookup wire ★
1 200mm length of 20-way ribbon cable
1 600mm length of 10-way ribbon cable
1 45 x 50mm sheet Presspahn or similar insulating material
1 40 x 45mm sheet of aluminium, 1.5-2.5mm thick
2 10 x 20mm sheets of aluminium, 1.5-2.5mm thick
1 90 x 70mm x 3mm thick sheet of clear acrylic/Perspex
★ all can be stripped from a 1m length of mains flex or a discarded mains cord

Parts list for CPU assembly

1 double-sided PCB coded 01106193, 60.5 x 62.5mm
1 2-way mini terminal block, 5.08mm spacing (CON5; optional)
2 5x2 pin headers (CON7, CON9-CON11, CON23)
1 10x2 pin header (CON8)
2 3-pin headers (LK1, LK2)
1 2-pin header (JP5)
3 shorting blocks (LK1, LK2, JP5)
1 ferrite bead (FB12)
1 miniature 8MHz crystal (X2) **OR**
1 standard 8MHz crystal with insulating washer (X2)
1 10kΩ vertical trimpot (VR1)

Semiconductors

1 PIC32MZ2048EFH064-250I/PT 32-bit microcontroller programmed with 0110619A.HEX, TQFP-64 (IC11)
1 25AA128-I/SN I²C EEPROM, SOIC-8 (IC12) #
1 LD1117V adjustable 800mA low-dropout regulator, TO-220 (REG2) #
1 LM317T adjustable 1A regulator, TO-220 (REG3)
1 blue SMD LED, SMA or SMB (LED2)

3 SGL41-40/BTM13-40 or similar 1A schottky diodes, MELF (MLB) (D14-D16)

Capacitors

1 470μF 10V electrolytic
5 10μF 50V electrolytic
11 100nF SMD 2012/0805 50V X7R
4 20pF SMD 2012/0805 50V C0G/NP0

Resistors (all SMD 2012/0805 1%)

1 10kΩ	1 1.2kΩ	2 1kΩ
1 560Ω	2 470Ω	1 390Ω
2 330Ω	1 100Ω	3 47Ω

Parts list for LCD assembly

1 128 x 64 pixel graphical LCD with a **KS0107/KS0108** controller and 20-pin connector
1 double-sided PCB, coded 01106196, 51 x 13mm
1 10x2 pin header
1 20-pin header

Parts list for front panel interface

1 double-sided PCB coded 18107212, 74.5 x 23mm
2 right-angle PCB-mount rotary encoders with inbuilt pushbuttons (RE1, RE2)
[Mouser [858-EN11-VSM1BQ20](#)]
2 right-angle PCB-mount sub-miniature momentary pushbutton switches (S1, S2)
1 5x2-pin IDC box header (CON1)
7 22nF 50V ceramic capacitors
2 10kΩ 1/4W 1% thin film axial resistors

Parts list for one regulator assembly (double the quantities for two)

1 double-sided PCB coded 18107211, 116 x 133mm
1 220μH 5A ferrite-cored toroidal inductor (L1)
1 10μH 6.6A ferrite-cored toroidal inductor (L2)
[Bourns **2000-100-V-RC**]
1 330μH 3A ferrite-cored toroidal inductor (L3)
(only needed for one module)
1 10A slow-blow M205 fuse (F1)
2 M205 PCB-mount fuse clips (for F1)
3 2-way screw terminals, 5.08mm pitch (CON1, CON2, CON4)
1 5x2-pin vertical header (CON3)
2 3-pin vertical polarised headers with matching plugs housings and pins (**optional** – for manual control) (CON5, CON6)
1 2-way vertical polarised header (CON7)
2 3-way pin headers with jumper shunts (JP1, JP2)
2 micro-U flag heatsinks (for REG1 & REG2)
6 TO-220 silicone insulating kits (washers and bushes)
4 15mm-long M3-tapped Nylon spacers
9 M3 x 16mm panhead machine screws
4 M3 x 6mm panhead machine screws
9 M3 hex nuts
13 flat washers, ~3.2mm ID (to suit M3 screws)
13 shakeproof washers, ~3.2mm ID (to suit M3 screws)

Semiconductors

1 INA282AIDR bidirectional current shunt monitor, SOIC-8 (IC2) #
1 LM358 dual single-supply op amp, DIP-8 (IC3)

Intelligent Dual Hybrid Supply part 1

1 MCP4922-E/P dual 12-bit digital-to-analog converter, DIP-14 (IC4) #
1 MCP3202-BI/P dual 12-bit analog-to-digital converter, DIP-8 (IC5) #
2 MAX14930EASE+ 4-channel isolators, SOIC-16 (IC6,IC7) #
2 LM317 1.5A adjustable linear regulators, TO-220 (REG1,REG2)
1 LM2575T-5.0V 5V 1A buck regulator, TO-220-5 (REG3) **(only needed for one module)**
1 LM337 1.5A adjustable negative regulator, TO-220 (REG4)
1 MC34167TV 0-40V 5A integrated buck regulator, TO-220-5 (REG5) #
2 LM1084IT-3.3 5A low-dropout regulators, TO-220 (REG6,REG7) #
2 BD139 80V 1A NPN transistors, TO-126 (Q3,Q10)
7 BC546 80V 100mA NPN transistors, TO-92 (Q4-Q8,Q11,Q13)
2 BC556 80V 100mA PNP transistors, TO-92 (Q9,Q12)
1 400V 10A bridge rectifier with metal base (BR1) **[eg, Compchip MP1004G-G] #**
9 1N4004 400V 1A diodes (D1,D2,D5,D6,D9,D10,D13,D17,D19)
1 6TQ045-M3 45V 6A schottky diode, TO-220AC (D3) #
1 P600K (or -M) 6A 800V diode (D8)
1 1N5819 40V 1A schottky diode (D12)
1 1N4148 signal diode (D14)
1 6.8V 400mW zener diode (ZD2) **[eg, 1N754]**

Capacitors

3 4700µF 50V 105°C electrolytic, 10mm pitch, ≤20mm diameter **[eg, Nichicon UVZ1H472MRD]**
1 3300µF 50V electrolytic
3 1000µF 50V low-ESR electrolytic
1 1000µF 50V electrolytic ≤13mm diameter
1 470µF 25V low-ESR electrolytic
2 220µF 50V low-ESR electrolytic
5 100µF 50V low-ESR electrolytic
2 15µF 50V solid tantalum, SMD E-case **[eg, Mouser 581-TPSE156M050H0250 or 80-T495X156M50ATE200]**
7 10µF 50V 105°C electrolytic
1 1µF 63V MKT
3 470nF 50V X7R SMD ceramic, M3216/1206-size
12 100nF 63V MKT
10 100nF 50V X7R multi-layer ceramic
2 100nF 50V X7R SMD ceramic, M2012/0805-size
1 1nF 50V X7R multi-layer ceramic

Resistors (1/4W 1% thin film axial unless otherwise stated)

2 180kΩ	5 1.8kΩ
1 15kΩ	1 1.2kΩ
1 12kΩ	3 1kΩ
12 10kΩ	2 680Ω
1 6.8kΩ	2 220Ω
1 4.7kΩ	1 100Ω
2 3.3kΩ	2 68Ω
2 0.05Ω (50mΩ) 1% 1W shunts	
[TT Electronics OAR1R050FLF] #	
1 0.01Ω (10mΩ) 1% 1W shunt	
[TT Electronics OAR1R010JLF] #	
2 0Ω resistors or lengths of 0.7mm diameter tinned copper wire (LK1,LK2) (only needed for one module)	

[Available from Mouser, DigiKey, Farnell etc]

is energy stored in the inductor. The 'input side' of the inductor, the node where the MC34167 output connects to it, still has current flowing into it. But the MC34167 switch is off. As a result, this node tries to go negative.

The 'catch' diode (D3) clamps this to about -0.5V as it is a schottky type. During this phase, current continues to flow into the output capacitor, but the energy is supplied from the inductor's collapsing magnetic field.

There are a few important things to keep in mind when designing a buck regulator:

- The switching nodes (input, output, diode, input capacitors and ground traces between these) all see current switching at 72kHz. These pulses have very fast rise and fall times, which means we need to be conscious of induced voltages across pins and tracks and the potential for these pulses being coupled into other parts of the circuit and indeed itself.
- The switchmode regulator's output pin is switching between the full input rail and -0.5V very rapidly and is a significant source of EMI.
- The catch diode carries substantial current; the duty cycle depends on the output voltage and current. The worst case is with a low output voltage and high current, where this device carries much of the load.
- The output ripple is heavily influenced by the inductor and capacitor values.

The principal losses in a switchmode regulator of this sort are in the switch. The MC34167 has a maximum voltage drop of 1.5V at full current. The catch diode will drop 0.5V when it is conducting, and there are resistive and core losses in the inductor. These losses add to a few watts, representing more than 70% efficiency in the worst case, and closer to 90% for higher currents.

So the pre-regulator's function in this circuit is to efficiently drop the unregulated input voltage, ensuring that the linear regulators only ever need to drop about 5V. This way, we can draw 5A from the power supply without excessive dissipation in the final regulator stage.

The circuit around the pre-regulator (REG4) is very similar to an ON Semiconductor (OnSemi) application note, but with a couple of important differences.

The output voltage of the MC34167 is set by the feedback pin (pin 1). If this is below 5V, the device's duty cycle increases to drive the output voltage up. Conversely, if this is above 5V, the duty cycle decreases.

We have used 6.8kΩ and 1.2kΩ resistors in the feedback divider, which would normally set the output to 33V. ($5.05V \times [6.8k\Omega + 1.2k\Omega] \div 1.2k\Omega$). This is more than we need, and we need to drop this to keep it 5V above the linear regulator output.

This is done by Q9, a BC556 PNP transistor across the 6.8kΩ feedback resistor, in conjunction with the 4.7kΩ and 1kΩ resistors providing feedback from the overall power supply output.

The 4.7kΩ and 1kΩ resistors divide the voltage difference between the pre-regulator and linear regulator, and this voltage drives the BC556 transistor to act as a feedback amplifier.

When the pre-regulator's output is too low, the base-emitter voltage on the BC556 is less than 0.6V. The current source turns off, and the feedback to the MC34167 is reduced. When the pre-regulator's output is too high,

Constructional Project

the base-emitter voltage of the BC556 is more than 0.6V, and the current source turns on, generating 5V across the 1.2kΩ resistor and increasing feedback to the MC34167.

The 68Ω resistor sets the maximum current from this current source, limiting the current we inject into the MC34167 sense pin, so that under fault conditions, we do not damage it.

Note how we are using the 0.6V typical V_{BE} of the BC556 as the voltage reference to achieve a nominal 5V drop for the output regulator. This does vary a little with temperature and overall output voltage, but that does not matter. The pre-regulator will always deliver about 5V more than the linear regulator.

The MC34167 is well within spec being fed from rectified 25V AC (about 33V after BR1) with margin for an unloaded transformer and mains voltage variation, without asking the device to work beyond its specified range.

A bonus of using a switchmode pre-regulator is that at lower output voltages, the system will be able to deliver more current than it demands at its input. Our software allows for this.

Control and monitoring

Control and monitoring of the Intelligent PSU are via an SPI serial interface to each board. This allows access to the optically-isolated DAC and ADC chips. These are both two-channel devices that allow programming of the output voltage and output current limit (via the DAC), and monitoring of the actual output voltage and current (via the ADC).

These digital signals are carried

over a 10-wire interface back to the control board, with the pinout shown in Table 1.

To increase versatility for situations where microprocessor control is not required, we have made provision for external potentiometers to set the voltage and current limit (via CON5 & CON6). If you choose to use this, simply leave off all components in the optically isolated section and also leave off the ADC and DAC chips.

The protocol for this interface is straightforward. Digital values are written to the DAC to set the voltage and current output and limits, and digital values are read from the ADC. If “rolling your own” interface, the panel opposite will be helpful.

ADC and DAC

The dual 12-bit ADC and dual 12-bit DAC are Microchip MCP4922 and MCP3202 devices respectively. Their very simple digital interfaces are described in their data sheets.

Calibration is required to convert the digital values, to and from voltages and currents. Our supplied control code handles this.

The isolation devices allow one microcontroller module to control and monitor multiple independent regulator modules, which could have their grounds connected to different potentials (via the output connectors).

There are two links, LK1 & LK2, that allow power to be fed back from one of the regulator modules to the control interface. If you are using the recommended microcontroller, then you install these on one, and only one, regulator module. It does not matter which.

Table 1 – control connector pinout

Pin	Function	Comment
1	DAC #1 chip select	Active Low
2	SPI SDO (to micro)	Also known as MISO
3	ADC #1 chip select	Active Low
4	SPI SDI (from micro)	Also known as MOSI
5	DAC #2 chip select	Active Low
6	SPI SCK (from micro)	Micro is SPI master
7	ADC #2 chip select	Active Low
8	SPARE	
9	GND	
10	V_{DD}	Either from micro or supplied to micro – see text

This allows the LM2575 regulator on that board to power the micro.

It also connects the microcontroller to the ground of this regulator module, but that is fine, as both will float together, but separately from the other regulator module.

The 12-bit devices have 4096 voltage steps. The linear output regulator compares the DAC voltage to the output voltage divided by 16 ($15k\Omega \div 1k\Omega + 1$). This means that the output voltage is controlled in 19.5mV steps ($5.0V \times 16 \div 4095$).

The INA282 IC which monitors the output current through the 10mΩ resistor includes 50 times amplification. So the full-scale output of the INA282 is 2.5V ($5A \times 0.01\Omega \times 50$), and this translates into an ADC measurement resolution of 2.4mA ($1A \times 0.01\Omega \times 50 \times [5V \div 2.5V] \div 4095$).

For setting the current limit, the DAC will have the same notional current per bit. The user interface software includes calibration for all these settings and measurements, so you do not need to install precision parts when building this.

The remainder of the circuit

The AC from the transformer is rectified by 10A bridge rectifier BR1. Above 3A, this will need heatsinking, so it is mounted on the heatsink via flying leads. There is the provision to mount it on the PCB for lower-current applications.

There is also a negative rail generator comprising diodes D5 & D6 and two capacitors, 3300μF & 1000μF. Using these values avoids output transients after switch-off. This generates a neg-

Table 2: resistor colour codes

Qty	Value	4-band code	5-band code
2	68Ω		
1	100Ω		
2	220Ω		
2	680Ω		
3	1.0kΩ		
1	1.2kΩ		
5	1.8kΩ		
2	3.3kΩ		
1	4.7kΩ		
1	6.8kΩ		
12	10kΩ		
1	12kΩ		
1	15kΩ		
2	180kΩ		

Intelligent Dual Hybrid Supply part 1

active rail for the op amps, so that the output voltage can go down to 0V, as described earlier. That negative rail is then fed to REG4 to produce the regulated -4.5V supply.

There are three 4700µF 50V capacitors for bulk storage, close to the switch-mode regulator. This is required to support the expected ripple current and to provide a very low-impedance supply to that regulator. Lower value capacitors can be used, but the maximum output voltage will be reduced.

There are two 15µF surface-mount tantalum capacitors on the top side of the board, and 470nF and 100nF SMD ceramics on the underside. These are located near the power and ground pins of the MC34167, to ensure that the MC34167 supply has a low source impedance at high frequencies. This minimises the chance of voltage spikes being induced in the power supply tracks.

The 50V ratings on these parts are for a good reason; as we've written previously, ceramic capacitors with higher voltage ratings perform better even when charged to lower voltages.

We have three 1000µF 50V low-ESR electrolytic capacitors in the output filter, in parallel with 470nF ceramic capacitors; these must handle the ripple current at 5A output. The output voltage is filtered again with a 10µH inductor & a 100µF low-ESR capacitor.

There are four other ancillary regulators on the board, none of which are configured unusually:

- +12V (11.5V actual) rail generated by REG1 (LM317), for the op amps.
- +5V (5.1V actual) rail, generated by REG2 (LM317) from the +12V rail, for the ADC and DAC chips.
- -4.5V (-4.5V actual) rail, generated by REG4 (LM337), for the op amps.
- +5V rail generated by REG3 (LM2575-5), a second switch-mode regulator which supplies the control interface, and optionally the microcontroller/user interface. An efficient switch-mode regulator is used here to allow the control interface to draw several hundred milliamps without creating much extra heat.

Transformer selection

The ideal transformer is a 300VA unit with two independent 25V AC

Controlling the Regulator Module via SPI

A DAC write is used to set the output voltage (channel 1) and the current limit (channel 2). First, drive chip select ($\overline{CS}$) low for the selected DAC. Then write 0x7000 (28,672 decimal) + 0x0 to 0xFFFF (4095 decimal) as the DAC value for the desired voltage. Or write 0x9000 + 0x0 to 0xFFFF to set the current limit. After the write, bring $\overline{CS}$ high again.

For example, to set the output to 5.1V: drive the DAC's $\overline{CS}$ low, send 601 to channel 1 (so write 0x7259), then take $\overline{CS}$ high again. Remember that many microcontrollers require you to read the SPI buffer after you write an SPI word.

To read the actual voltage and current for each channel, you need to query the ADC. Keep write speeds reasonable; we have used 100kHz, which allows good accuracy on the ADC, and provides easy setup and hold times.

Drive $\overline{CS}$ low for the selected ADC, then send the read command byte: 0x01. Make sure you wait until the whole SPI byte has been sent from your micro to the ADC, then read a byte and discard it.

Next, send the read command 0xA0 for voltage, or 0xE0 for current. Make sure you wait until the whole SPI byte has been sent from your micro to the ADC, then read and store the next byte. Write 0x00 to the ADC, wait until the whole SPI byte has been from your micro to the ADC, then read another byte.

The last byte read contains the lower 8 bits of the result, while the upper 4 bits of the 12-bit result are in the lower 4 bits of the previous byte read. So, for example, in the C language you can compute the read value as:

```
unsigned short value = (byte1 & 0x0F)*256 + byte2;
```

secondary windings. We used the Altronics M5525C, a 25+25V AC, 300VA transformer. This design is very versatile and will happily operate from anything above 15V. The only essential feature is that the secondary windings are not internally joined.

Farnell has a couple of transformers that look like they would fit the bill nicely. The Multicomp Pro [VTX-146-300-125](#) transformer is £41 at the time of writing and it has a 230V AC primary that would suit the UK/European mains supply.

They also sell the Multicomp Pro [MCTA300/25](#) for £52.58. It is the same size, with a dual primary, configurable for 115/230V AC. Still, if your mains supply is 220-240V AC, you might as well stick with the slightly cheaper option.

We have set a current limit for the power supply at 5A per rail and a maximum output voltage of 25V DC. It is important that when you set up the controller that you enter the correct VA rating for the transformer, and its nominal AC voltage. These are used to calculate current limits that are used to protect the transformer from being overloaded.

Control circuit

This control circuit has been used in several previous projects, starting with the DSP Active Crossover & 8-channel

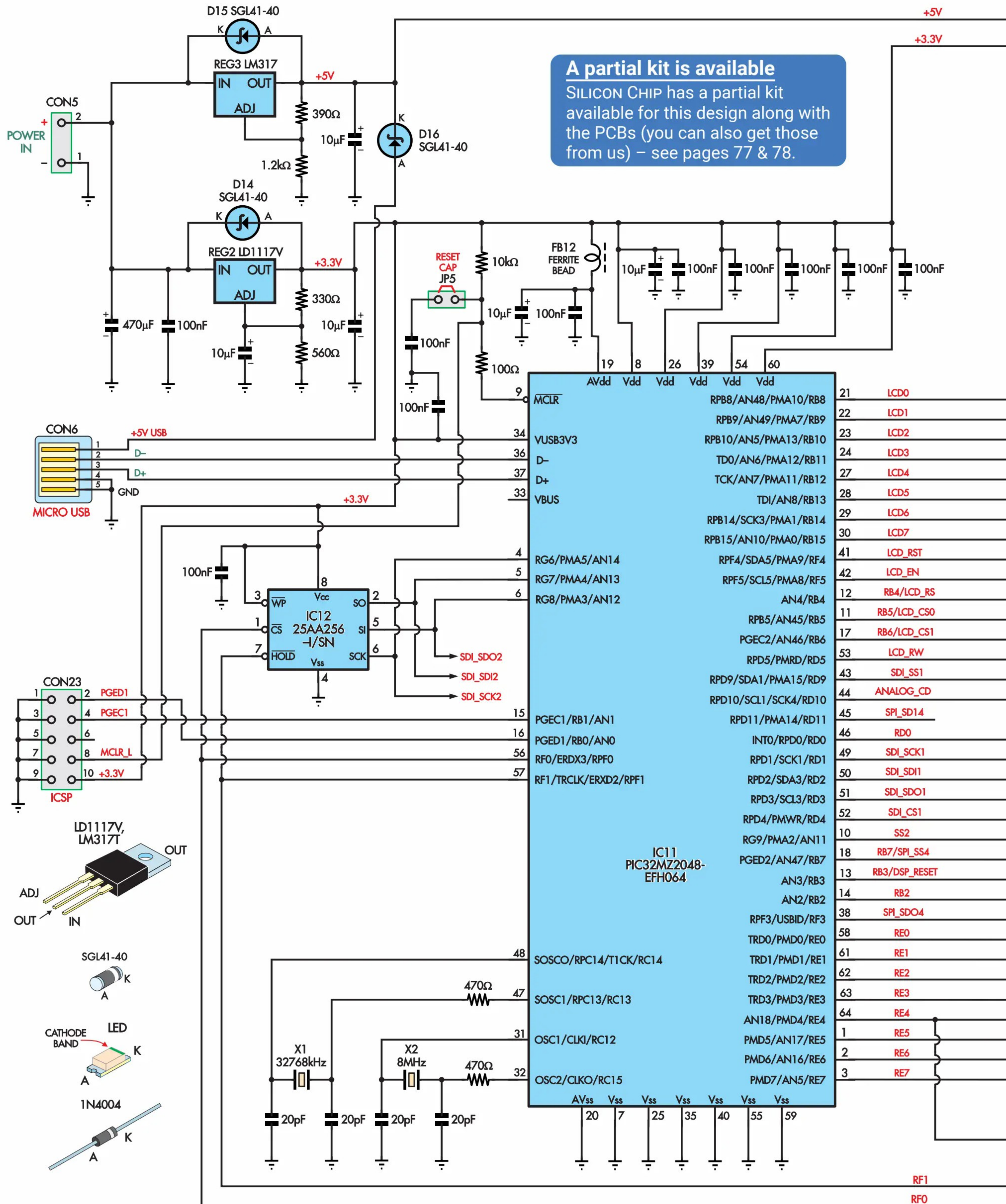
Parametric Equaliser. That series was published in the January, February & March 2020 issues of *Practical Electronics*. As in that project, the interface is displayed on a monochrome graphical LCD. That LCD, the front panel control board and the regulator boards are wired back to the control board via ribbon cables and multi-pin headers.

The control circuit is reproduced here; see Fig.8. Microcontroller IC11 is a PIC32MZ2048 32-bit processor with 2MB flash and 512KB RAM, which can run up to 252MHz. It has a USB interface brought out to a micro type-B socket, CON6, although we haven't used it in this project – it's there 'just in case' for other projects.

The PIC is also fitted with an 8MHz crystal for its primary clock signal (X2). Provision is made on the PCB (and shown in the circuit) for a 32.768kHz crystal for possible future expansion, but it is not used in this project and can be left out. There is also a serial EEPROM which is used to store the calibration values, voltage and current settings. This must be fitted.

The front panel controls are wired back to 10-pin header CON11 (and on to PORT E of the micro). The regulator board(s) connect to 10-pin header CON7. The other headers and connectors are unused in this project. 5V

Constructional Project



Intelligent Dual Hybrid Supply part 1

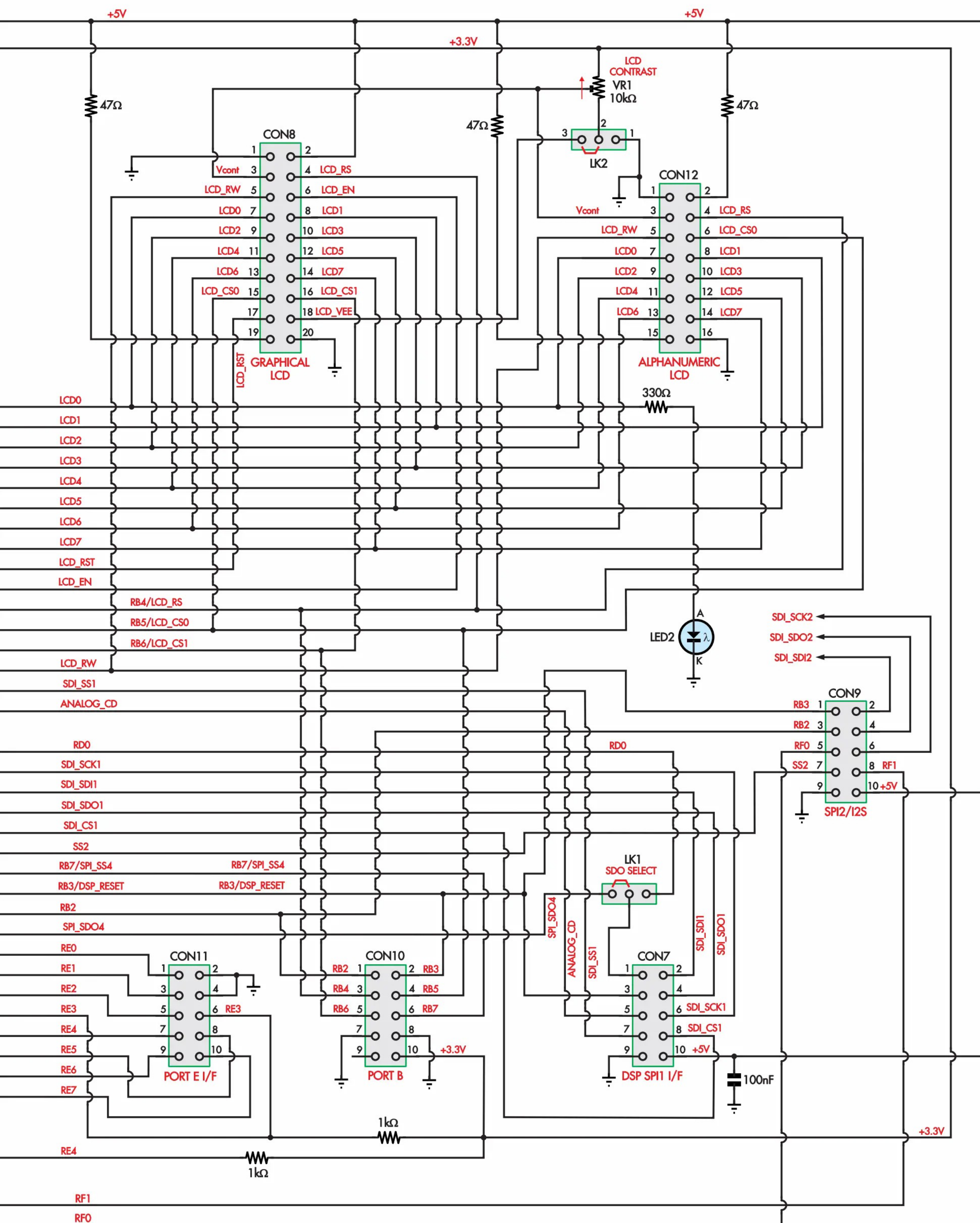


Fig.8: this CPU control circuit has been used in several projects. It includes a powerful 32-bit PIC32MZ processor, an 8MHz core crystal, an optional 32768Hz timekeeping crystal, 5V & 3.3V regulators, an SPI EEPROM, plus numerous connectors. The timekeeping crystal and 5V regulator are not needed for this project. CON7 connects to the regulator boards, CON11 to the front panel control board and CON12 to the LCD; the other connectors are unused.

Constructional Project

power for this board is applied across pins 10 & 9 of CON7 from one of the regulator boards.

The user interface is displayed on a graphical LCD, wired up to CON8 on the micro board via a ribbon cable. This provides a reasonably standard 8-bit parallel LCD drive interface. The eight LCD data lines (DB0-DB7) are driven from a contiguous set of digital outputs of IC11 (RB8-RB15). This allows a byte of data to be transferred to the display with just a few lines of code and minimal delay.

The other LCD control lines are driven by digital outputs RB4, RB5, RB6, RD5, RF4 and RF5 and the screen is powered from the 5V rail, with the backlight brightness set with a 47Ω resistor. The LCD contrast is adjusted using trimpot VR1, which connects to CON8 via LK2.

CON23 is a somewhat unusual in-circuit serial programming (ICSP) header. It has a similar pinout to a PICkit 3/4 but not directly compatible; it's designed to work over a longer cable. Since each signal line has at least one ground wire between it, signal integrity should be better.

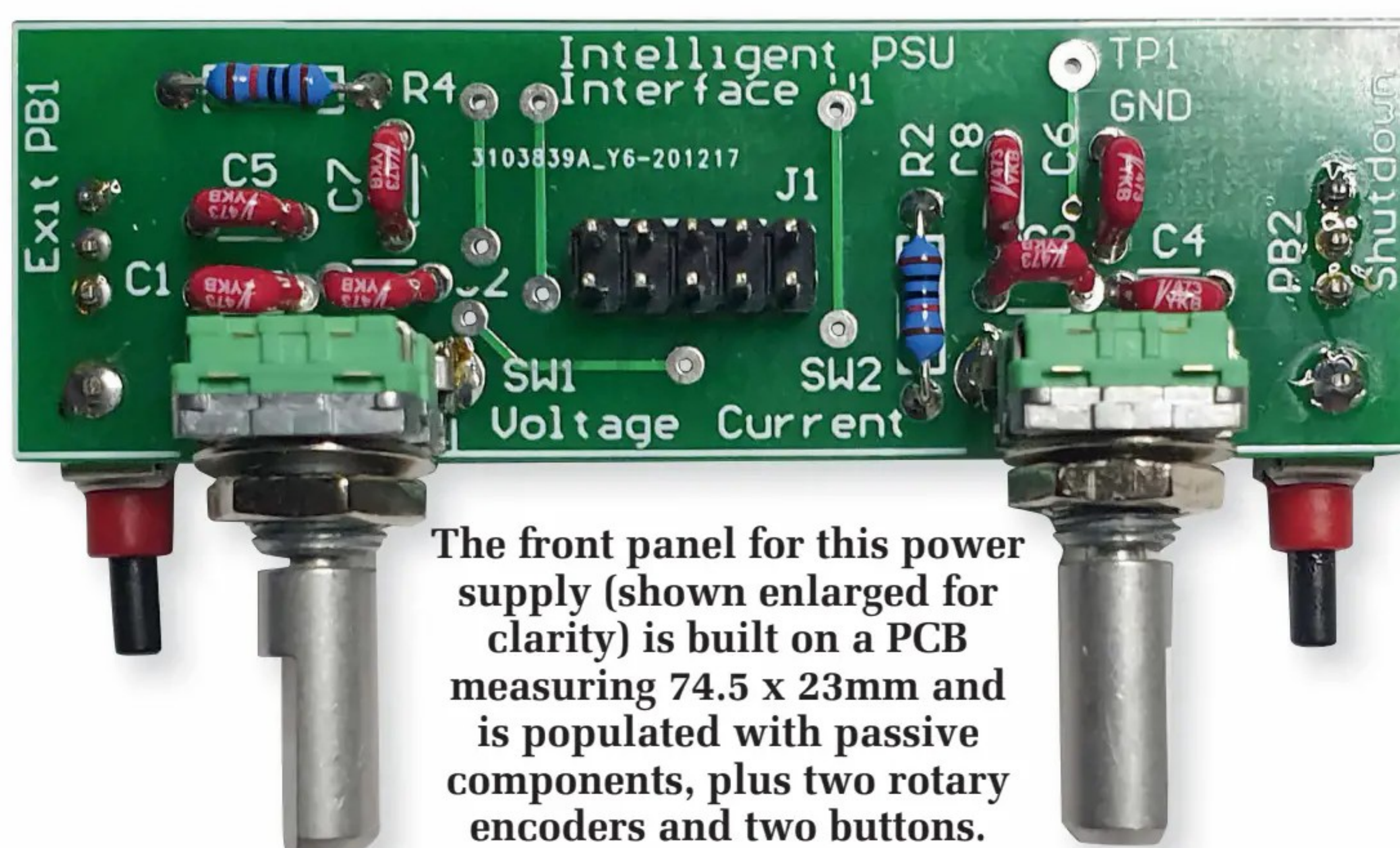
Jumper leads could be used to make a quick connection to a PICkit to program the microcontroller the first time. Or you could attach a 10-pin IDC connector to the end of a ribbon cable and then solder the appropriate wires at the other end of the cable to a 5-way SIL header as a more permanent programming adaptor for development use.

There are two regulators on the board, but REG3 is not needed in this case because the 5V rail is generated on the regulator board. REG2 is required, though, to produce a +3.3V rail from the 5V rail via schottky diode D15, powering microcontroller IC11.

LED2 is connected from LCD data line LCD0 to ground, with a 330Ω current limiting resistor, so it will flash when the LCD screen is being updated.

Front panel board

Fig.9 shows the circuit of the front panel board specific to this project, and there isn't a whole lot to it. Rotary encoders RE1 & RE2 generate "Gray codes" by closing switch contacts between pins 1 & 3 and pin 2 (common). They also have integrated pushbutton switches that connect pins 4 & 5 when pressed, plus there are two separate momentary pushbutton switches, S1 & S2.



The front panel for this power supply (shown enlarged for clarity) is built on a PCB measuring 74.5 x 23mm and is populated with passive components, plus two rotary encoders and two buttons.

All these switch contacts have 22nF debouncing capacitors across them; there might not appear to be one across switch integrated into RE2, but it is in parallel with the other one, so they share one debouncing cap.

The Gray code outputs of rotary encoder RE2 have pull-up resistors, while those of RE1 do not, because the micro can provide pull-up currents on those pins.

All the switch contacts are wired either between a micro pin and GND, or a micro pin and the +3.3V rail, depending on what's most convenient

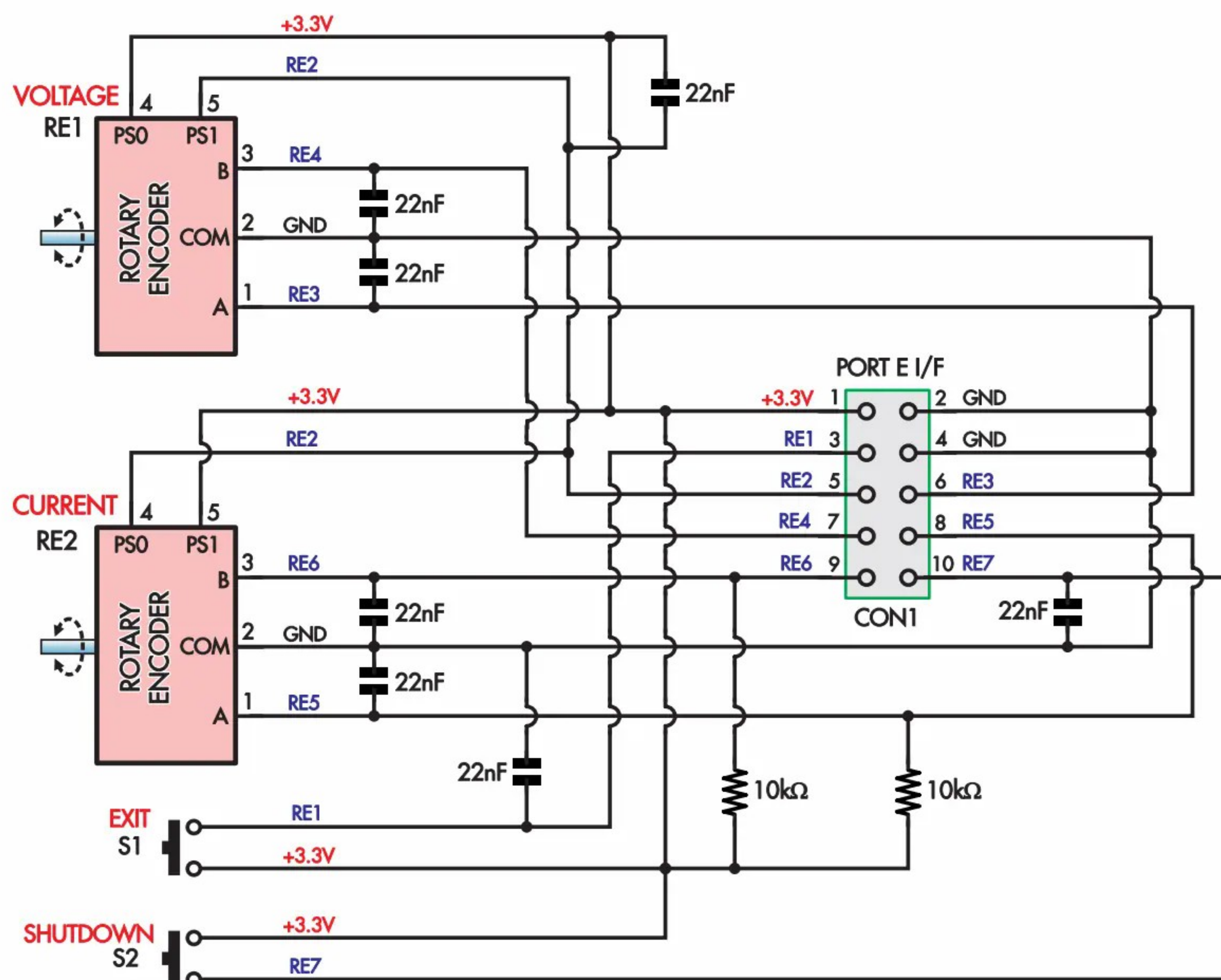
for the software to deal with. Those connections go back to the micro pins via CON1.

Next month

We have finished describing how the Intelligent PSU operates. Next month, we will present the details of the three main PCBs, describe how to assemble them, mount them in the case, and wire up and test the unit.

We'll also show you how to use the device and control it via the LCD graphical interface and front panel controls.

PE



SC
©2021

INTELLIGENT POWER SUPPLY

FRONT PANEL PCB CIRCUIT

Fig.9: the front panel circuit includes two rotary encoders with integrated pushbutton switches, plus two extra buttons and a handful of debouncing capacitors. 10-pin header CON1 on this board is wired back to CON11 (in Fig.8), so the micro can sense when the encoders are rotated and buttons are pressed.



Practical Electronics

UK readers **SAVE £1** on every issue

SUBSCRIBE NOW!

01202 087631



Take out a one-year subscription and save more than **£10** over the year.

Even better – save over **£1** per issue if you subscribe for two years – a total saving of **£26**.

Overseas rates represent exceptional value

You also:

- Avoid any cover price increase for the duration of your subscription
- Get your magazine delivered to your door each month
- Guarantee a copy, even if your local newsagent sells out

Order by:

- Phone or post with a cheque, postal order or credit card
- Or order online at: www.electronpublishing.com



Practical Electronics

SUBSCRIPTION PRICES

Subscriptions for delivery direct to any address in:

UK: 6-months £33.99, 12-months £59.99, 24-months £114.99

Europe Airmail: 6-months £38.99, 12-months £69.99, 24-months £129.99

Rest Of The World Airmail: 6-months £46.99, 12-months £84.99, 24-months £164.99

Cheques (in **£ sterling only**) payable to **Select Publisher Services Ltd** and sent to: Practical Electronics Subscriptions, PO Box 6337, Bournemouth BH1 9EH, United Kingdom

Tel: 01202 087631

Email: pesubs@selectps.com

Also via our online shop at: www.electronpublishing.com

Subscriptions start with the next available issue. We accept MasterCard or Visa.

(For past issues see the Back Issues page.)



DIGITAL SUBSCRIPTIONS

Electronic subscriptions are available from £29.99 for 12 months, more details at: www.electronpublishing.com

SUBSCRIPTION ORDER FORM

- ☐ 6 Months: UK £33.99, Europe £38.99 (Airmail), Rest of the World £46.99 (Airmail)
- ☐ 1 Year: UK £59.99, Europe £69.99 (Airmail), Rest of the World £84.99 (Airmail)
- ☐ 2 Years: UK £114.99, Europe £129.99 (Airmail), Rest of the World £164.99 (Airmail)

To: Practical Electronics Subscriptions
PO Box 6337
Bournemouth BH1 9EH
United Kingdom

Tel: 01202 087631

Email: pesubs@selectps.com

I enclose payment of **£**
(cheque/PO in £ sterling only)

payable to **Select Publisher Services Ltd**

☐ Please charge my Visa/Mastercard



My card number is:
Please print clearly, and check that you have the number correct

Card Security Code Valid From Date.....
(The last 3 digits on or just under the signature strip)

Card Ex. Date

Name

Address

Post code Tel.

Email

Subscriptions start with the next available issue.

Circuit Surgery

Regular clinic by Ian Bell

Topics in digital signal processing – Implementing DSP on a microcontroller, part one

We are looking at various topics related to digital signal processing (DSP).

DSP covers a wide range of electronics applications where signals are manipulated, analysed, generated, stored or displayed as digital data but originate from and/or are converted to real-world signals for interaction with humans or other parts of the physical world.

Fig.1 shows the key elements of a generic DSP system with a signal path from an analog input via digital processing to an analog output. This does not necessarily represent every DSP system (not all have all the parts shown), but it serves as a reference for the various subsystems we will look at.

In our discussions on DSP so far, we have covered concepts relating to all the stages in the generic DSP system in Fig.1.

The earlier articles focused on sampling, filtering and data conversion. Many of the LTspice simulations used in them to illustrate the ideas discussed can be easily related to real circuit implementations, even if we used idealised components such as the behavioural sampler and filters provided in the LTspice library.

Our more recent discussions on digital filters have also used LTspice simulations, but these are more abstract – effectively just performing the filter maths in LTspice so we can use LTspice's analysis functions to plot the filters' responses. Real implementations require some additional considerations, which is what we will be looking at next.

There are two fundamental ways of implementing a DSP function such as filtering: directly in digital hardware, or in software. We will look briefly at hardware, cover some general topics related to implementing arithmetic operations and then discuss software implementation

on a microcontroller in more depth, including an example filter.

Digital filtering in hardware

For the FIR filters we have been discussing, we can build a digital circuit based on the structure shown in Fig.2. This is the same structure used in recent articles, except that the addition operations are shown as a single block. The Δt delay blocks are implemented as a shift register, while the multiply and add blocks are implemented using multiplier and adder arithmetic circuits.

The coefficients (a_i) can be held in registers (allowing them to be changed), or hard-wired as fixed values if the filter has a fixed response in the application.

The shift register is fed by the input data ($x[n]$) and comprises a set of registers (the Δt blocks), each holding one sample, so each register has the number of bits needed to represent one sample value. The shift register is clocked at the sampling rate, so it presents a new set of data to the multipliers and adders in each cycle, which in turn produce the current output value ($y[n]$).

The multipliers and adders could be implemented as one large combinational (stateless) logic circuit; however, this must deliver the correct output value within one clock cycle (the sampling period). Large combinational logic circuits can be slow, which would restrict the maximum sample rate.

The speed is related to the number of logic gates in the longest path from input to output (assuming each gate has about the same delay). One solution to this is to insert registers into the combinational logic to reduce the path length. This is called pipelining.

Fig.3 shows an example arithmetic circuit (simpler than the filter). This adds

four values to give the result $R = D1 + D2 + D3 + D4$. Three combinational circuit adders ($\Sigma 1$ to $\Sigma 3$) are used in a typical 'tree' structure to add the four values.

The input data is loaded in registers R1 to R4 in a given clock cycle. The adders must deliver the result before the next clock cycle, at which point the result is loaded into register R5 at the same time as the next input data loads in R1 to R4.

Fig.4 shows a pipelined version of the circuit in Fig.3. Two additional registers (RP1 and RP2) are inserted between the adders. On the first clock cycle, after a given set of data is loaded into R1 to R4, the values $D1+D2$ and $D3+D4$ are loaded into RP1 and RP2, respectively. These intermediate results are input to adder $\Sigma 3$ to obtain the final result in the R3 register on the following clock cycle.

Thus, the circuit in Fig.4 takes two clock cycles to output the result, compared with the single cycle for the circuit in Fig.3. The number of clock cycles required to produce a result is referred to as the latency of the circuit.

The circuit in Fig.4 is faster than the one in Fig.3 because the longest path between registers is one adder rather than two. Therefore, its maximum clock speed is approximately twice that of the non-pipelined version. This means that although the clock cycle latency is

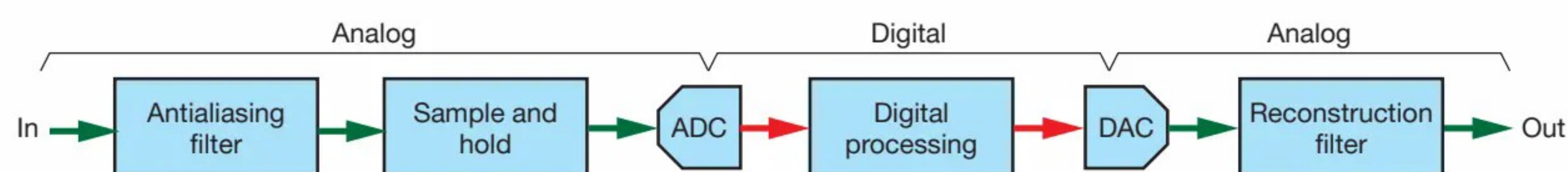


Fig.1: a generic digital signal processing (DSP) system structure.

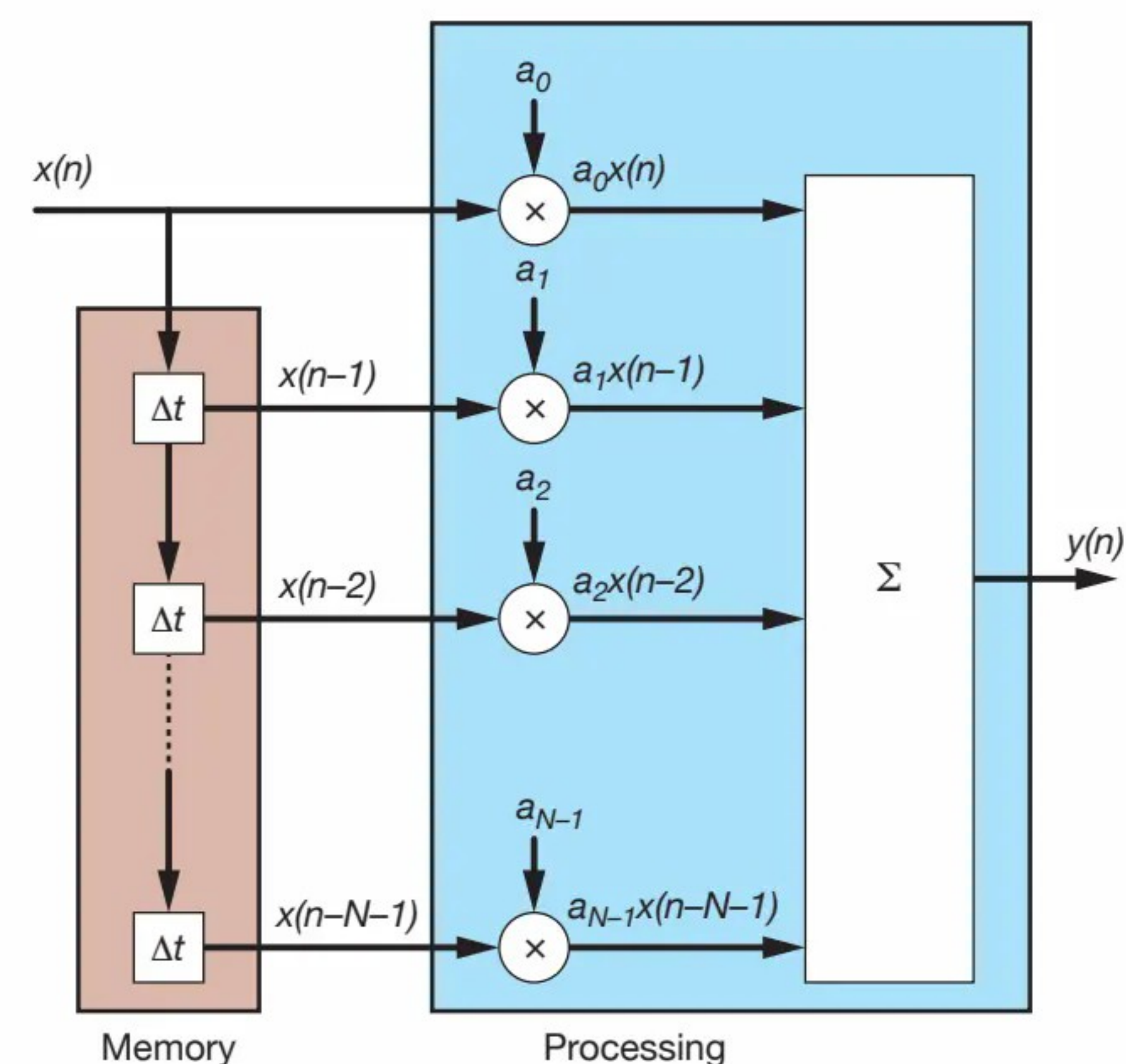


Fig.2: a direct-form FIR filter structure.

higher, it can process twice as much data per second. In DSP terms, it can operate at twice the sampling rate.

The structure in Fig.2 is called the direct form, or canonical form, of the FIR filter. It is the most straightforward to understand, but it is not necessarily the most effective in terms of circuit size and speed. There are various approaches available to reduce the size of the circuit structure and/or improve the speed, such as the pipeline technique described above.

Pipelining is widely applicable, but there are also more specific techniques for optimising the filter structures and arithmetic circuits they use. These are covered by a large body of academic literature and have names such as “common subexpression elimination” and “coefficient grouping”.

Editor’s Note: DSPs typically use ‘carry-lookahead adders’ (CLAs), which can sum multiple values more efficiently than the obvious approach. This and other ‘speed up’ techniques may avoid the need for pipelining.

CLAs can also be pipelined for greater speed. Larger, more complex designs, such as floating point arithmetic circuits, are more likely to need pipelining. The details of these advanced techniques are beyond the scope of this article.

Number formats

In both hardware and software DSP implementations, the way that numerical values are represented is important. For DSP in custom hardware, we have to build the circuit to match the chosen format.

In software it is usually a matter of declaring the appropriate data type in your code, but of course there is still hardware involved. The processor and its compatibility with a given number format can have a significant impact on performance.

Awareness of different number formats and their appropriateness for a particular task, together with what is best supported by any processor under consideration, is important when designing a DSP system. This is a topic which

can get very detailed, so we will just take a quick look at some of the basics.

The simplest numbers to deal with are positive whole numbers (unsigned integers), which are binary (base-2) numerical values. For example, decimal 3 is 0011 in 4-bit binary. For negative numbers, one bit can be used for the sign (ie, to indicate whether a number is positive or negative).

However, this approach does not work well when building integer arithmetic circuits, and has the complication of having two codes for zero (a positive and negative ‘zero’).

Two’s complement is usually used instead. The first bit still indicates the sign, but only code 0000 represents zero. Positive values up to $2^{N-1} - 1$ (for N bits) have codes which are the same as unsigned binary. Negative numbers have codes which are the same as that for the unsigned number obtained by subtracting the value from 2^N .

For example, in four-bits two’s complement, -2 has the same code as unsigned $16 - 2 = 14$, that is, 1110 in binary. To change the sign of a two’s complement number, you invert all the bits and add 1.

One major advantage of two’s complement is that standard adder circuits can be used with two’s complement numbers if the final carry out is ignored. The numbers move through zero nicely when you increment and decrement; for example 1111 is -1 in four bits. Adding one and ignoring the carry to the fifth bit gives 0000 (zero) as expected.

In sign magnitude format, 1111 is -7, so this does not work.

Fixed and floating point

DSP will often require use of real numbers (with a decimal point), not just integers. There are two major options for representing these in binary codes: floating point and fixed point.

Fixed-point numbers scale values so that they can be represented as integers. For example, an 8-bit code can represent unsigned integers from 0 to 255, but if we scale it by 8, it can represent values in steps of $\frac{1}{8}$, from 0.125 to 31.875. This means effectively the upper five bits are

used for the integer part (0 to 31) and the lower three bits for the fractional part (0 to $\frac{7}{8}$).

A binary word can be split in any proportion between the integer and fractional part. The Q notation is commonly used to define the format used; for example, Q5.3 means five bits for the integer and three for the fraction, as in the example above, although Q5.3 would usually imply a signed value (the same splitting approach works fine for two’s complement numbers).

Floating-point numbers are a form of scientific notation (eg, 1.23×10^4) represented in binary code. Several floating point formats have been used historically, but the one specified by the IEEE 754 standard is the most common.

The number has two key parts: the mantissa (or significand), which is a signed value that holds the digits of the number (1.23 in the example), and the exponent, which indicates the position of the decimal (or binary) point (the 4 in the example above).

A 32-bit IEEE 754 number is referred to as a single-precision value, while a 64-bit value is double precision. A single-precision value comprises a sign bit (S), an 8-bit exponent (E) and a 23-bit mantissa (M). The exponent is not signed, but is stored with an offset (bias) of 127. The value is given by:

$$(-1)^S \times 1.M \times 2^{(E-127)}$$

The “1.M” part is because mantissa always has an assumed first bit of 1, unless the exponent is at its minimum, so this bit is not stored. The separate sign bit means there are two zeros (+0 and -0).

An all-ones exponent is used for special cases, like errors leading to ‘not a number’ (NaN) results and infinity. This means the exponent range for normal numbers is from -127 to +127.

Floating-point numbers have a much larger dynamic range than fixed point, ie, the range from smallest to largest value representable with a given number of bits is much bigger than for fixed point. This is useful in DSP, as the values occurring in calculations may vary over large ranges.

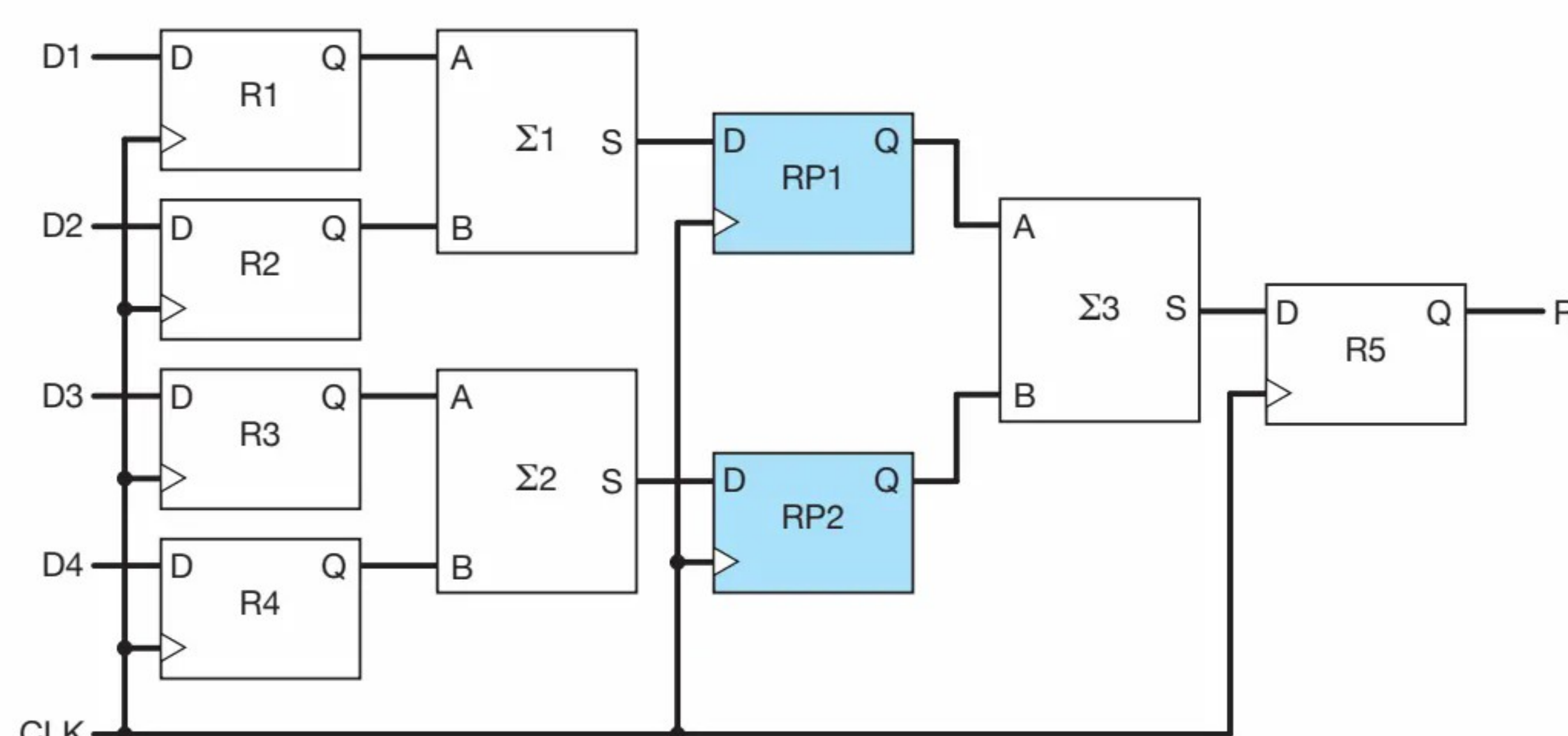
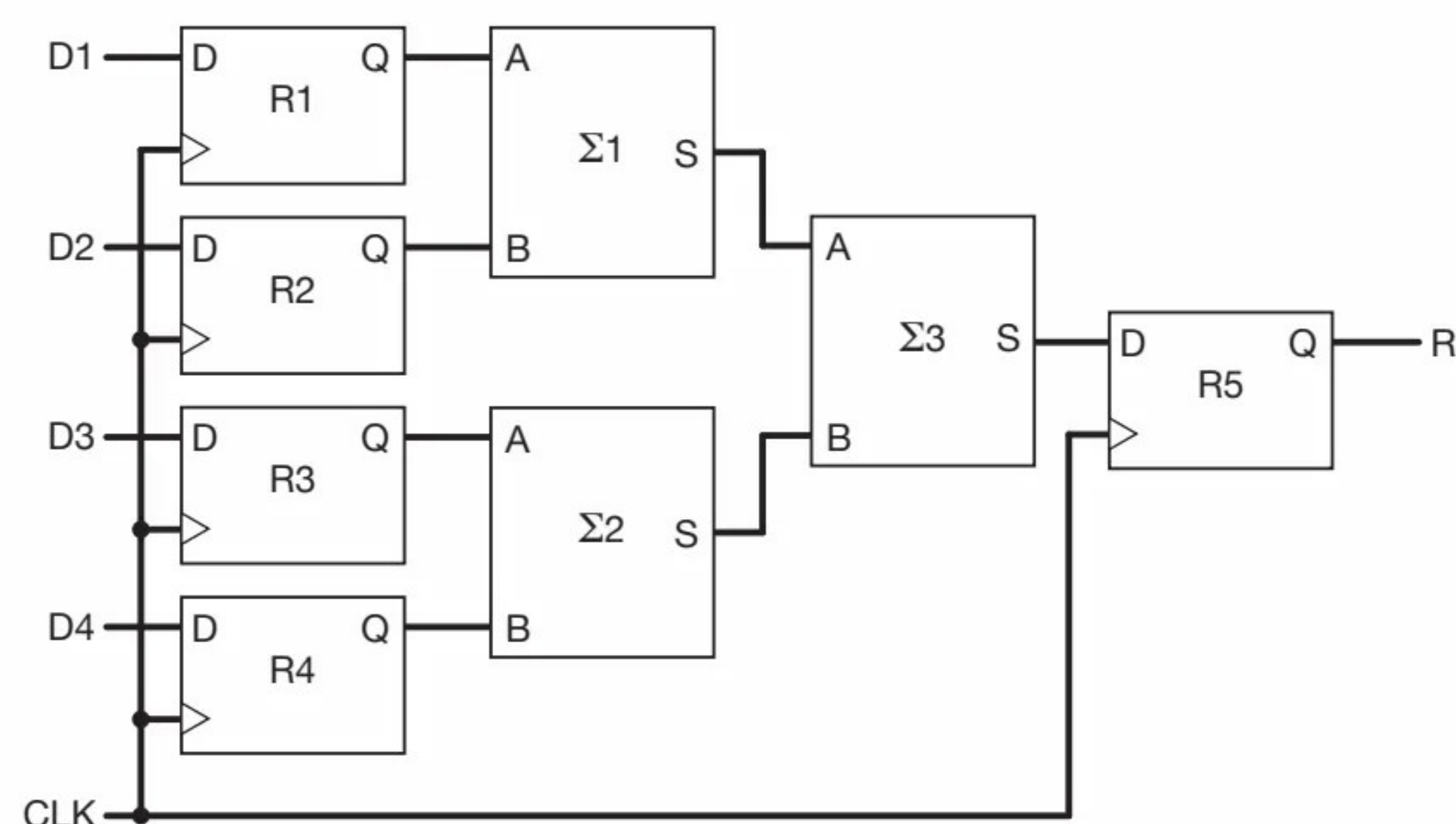


Fig.3: an example arithmetic circuit that sums four values. Fig.4: a pipelined version of the circuit in Fig.3 with added registers.

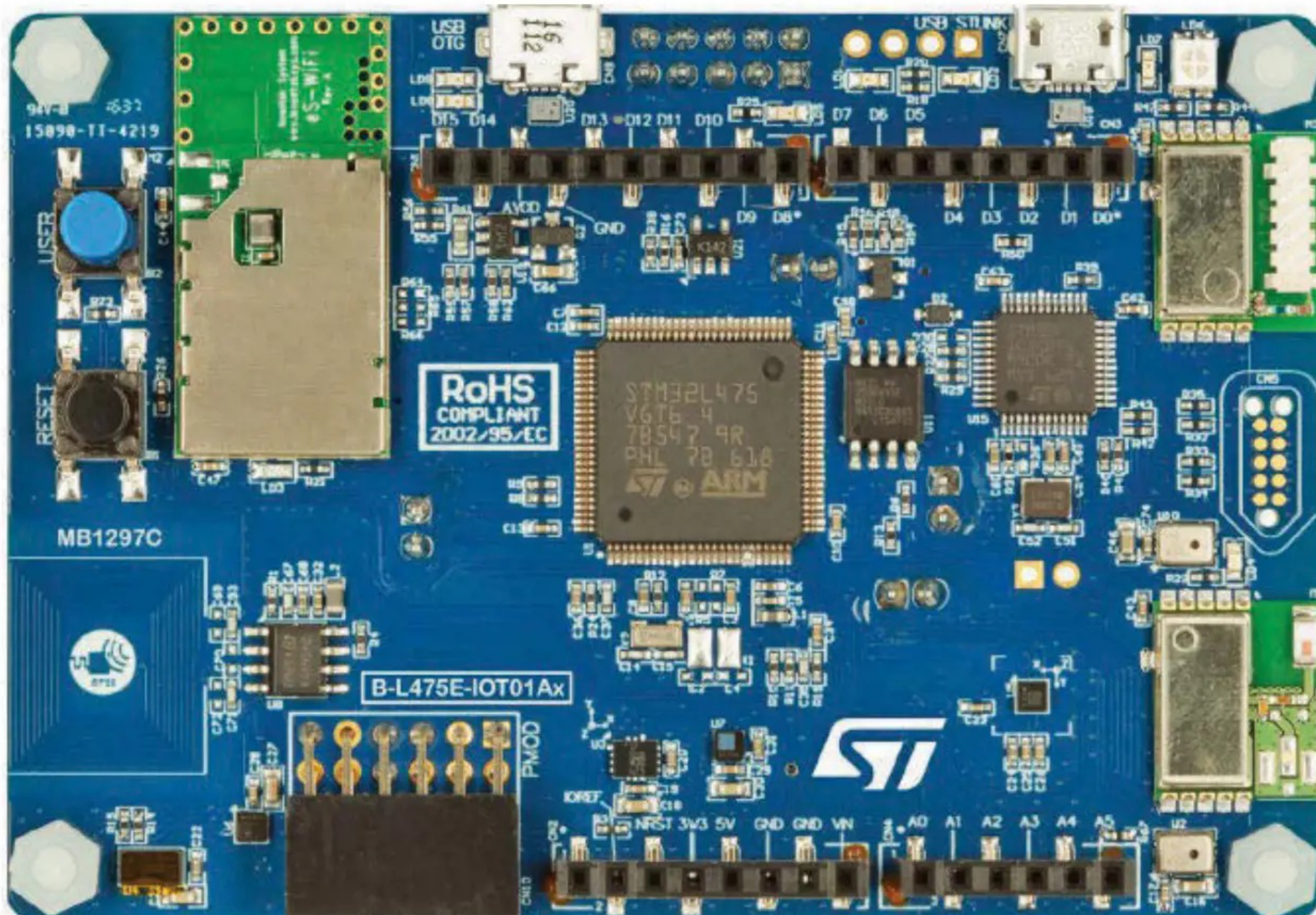


Fig.5: the STMicroelectronics B-L475E-IOT01A development board.

However, decimal values do not have exact representation in binary floating point. This, together with rounding errors, can lead to unexpected results. For example, a direct comparison of two floating-point numbers from different calculations will often evaluate as false due to very small differences, even if they are ideally equal. However, appropriate coding techniques can overcome this.

Adjacent fixed-point numbers (of a given format) always have the same difference between them, whereas gaps vary enormously with floating-point numbers, depending on the exponent. This means that floating point provides higher precision and smaller rounding errors over large value ranges.

Therefore, floating point is generally easier to use than fixed point for intensive calculations, as the latter may require format manipulations to help maintain precision. On the other hand, fixed-point is simpler where the dynamic range is known to be well constrained. Rounding and precision limitations create noise in DSP signals similar to the random noise inherent in analog circuits such as amplifiers.

DSP is maths-intensive, so it is often useful to employ floating point operations in the code. This is very slow if the floating-point maths is performed using the standard integer-based arithmetic hardware found in all processors. Processors that have floating-point hardware to perform operations like add and multiply achieve a significantly faster speed but tend to be more expensive.

Microcontrollers

The hardware approach to DSP requires either a custom IC (ASIC) design, which is only viable for major manufacturers,

or an FPGA (Field Programmable Gate Array) implementation. Low-cost FPGA development is possible, but the learning curve is steep if you have not worked with this technology before.

We will therefore concentrate on an example of a microcontroller-based software implementation.

Most readers will already be familiar with microcontrollers from the *Cool Beans* column and the numerous *Practical Electronics* projects that feature them. DSP can be implemented on just about any microcontroller; however, basic devices are very limited in what can be achieved.

Many microcontroller manufacturers make devices suitable for DSP applications and either name them specifically to indicate this (eg, dsPICs from Microchip) or make DSP suitability prominent in their marketing and datasheets. The following are examples of features which help make microcontrollers suitable for DSP:

- A high clock speed/instruction processing rate.
- A larger word size (eg, 32-bit rather than 8-bit processors).
- A hardware multiplier.
- A floating point unit (FPU), which implements floating-point operations (such as add and multiply) in hardware, providing calculation speeds that are orders of magnitude faster.
- A DSP instruction set. All processors have machine code instructions for basic arithmetic operations (such as add and subtract). DSP instructions extend the available operations to help facilitate efficient DSP code. Examples include multiply/accumulate and saturating addition (the result does not overflow but sticks at the maximum possible value). These can be used by writing in assembly language or some

compilers (eg, C/C++) can use them to create more optimal code.

- Internal cache memory. This is a small, fast memory that stores copies of frequently used memory data so the processor does not have to wait for memory access. All large processors have caches, but basic microcontrollers don't necessarily.
- Good on-chip ADCs and DACs and possibly other on-chip analog peripherals such as programmable gain amplifiers (PGAs). Systems requiring very high-performance ADCs/DACs often require separate devices, but on-chip is very convenient and facilitates experimentation with a relatively minimal development board. Most microcontrollers have ADCs, but DACs are less common.
- Direct memory access (DMA) controllers. These facilitate data transfer from/to peripherals such as ADCs and DACs while the CPU is busy doing calculations.
- Serial audio interfaces (SAIs) such as Inter-Integrated Circuit Sound (I²S) and Sony/Philips Digital Interface (S/PDIF).
- Software libraries supporting common DSP functions.

An example microcontroller

The example we will discuss uses a microcontroller from the popular STM32 family from ST Microelectronics (ST for short). These are based on the widely used Arm Cortex-M processor design. There are many devices in this family, which is divided into categories aimed at high performance, mainstream, ultra-low-power and wireless systems.

The specific chip used is the STM32L475 80MHz 32-bit Cortex-M4 processor, which is from the low-power category. Still, it ticks all boxes for the DSP features listed above. This chip comes on a development board, the B-L475E-IOT01A Discovery kit (see Fig.5), one of which I already had.

While suitable for DSP, the board is actually aimed at Internet of Things (IoT) applications (hence the low power microcontroller). The B-L475E-IOT01A has connectors that are compatible with the Arduino Uno V3, so most shields can be used with this board.

The STM32 ecosystem includes the wide variety of processors just mentioned, and a range of development boards from the minimal, small Blue Pill boards via Nucleo and Discovery to the feature-loaded Eval boards. The number of features, and consequently cost, generally increases in the order just listed.

The B-L475E-IOT01A board has a lot of features that are not required for our DSP example, so a simpler board with the same or similar processor would

suffice. The key requirements are an on-chip ADC and DAC, FPU and DMA. This month, we will discuss in general terms what is needed to set up the DSP project, provide some background on the STM32 ecosystem as well as giving some specific details for this board. Next month, we will look at aspects of coding the implementation in more detail.

As with many development boards, the microcontroller can be programmed (and debugged) from a PC or laptop via USB. The ST system for this is called ST-Link, and the hardware required is built into many, but not all, of the development boards (separate ST-Link programmers are available). In some cases, the boards are designed to allow the ST-Link part to be snapped/cut off if it is no longer needed.

Development tools

There are numerous software tools that can be used to write code for, program and debug STM32 devices. A straightforward option is to use the free Integrated Development Environment (IDE) provided by ST, which is called STM32CubeIDE. This allows you to develop C/C++ code and debug programs running on an attached development board.

STM32CubeIDE integrates another app called STM32CubeMX, which provides a graphical user interface for configuring the hardware on a chosen processor (see Fig.6) – the Device Configuration Tool.

We will mention key aspects of setting up the example DSP project, but will not provide complete step-by-step instructions. STM32CubeIDE is widely used, so tutorials that show the basics are available online (for example, <https://youtu.be/Hffw-m9fuxc>).

In general, if you do not know which processor or board might be suitable for a project, the IDE includes a part finder where you can search for suitable choices by defining the features you require.

Code generation

The IDE allows you to select a board from the list of available parts (those made by ST). It will initialise a project configured for the relevant board, after which you can customise the configuration via the Device Configuration Tool shown in Fig.6.

As with most IDEs, STM32CubeIDE works in terms of projects, based on folders in your file system. These folders will end up containing many files, most of which you do not need to look at, but it is important to maintain the integrity of the project structure used by the software.

On most microcontrollers, each I/O pin is connected to multiple on-chip peripherals, such as ADCs, serial bus

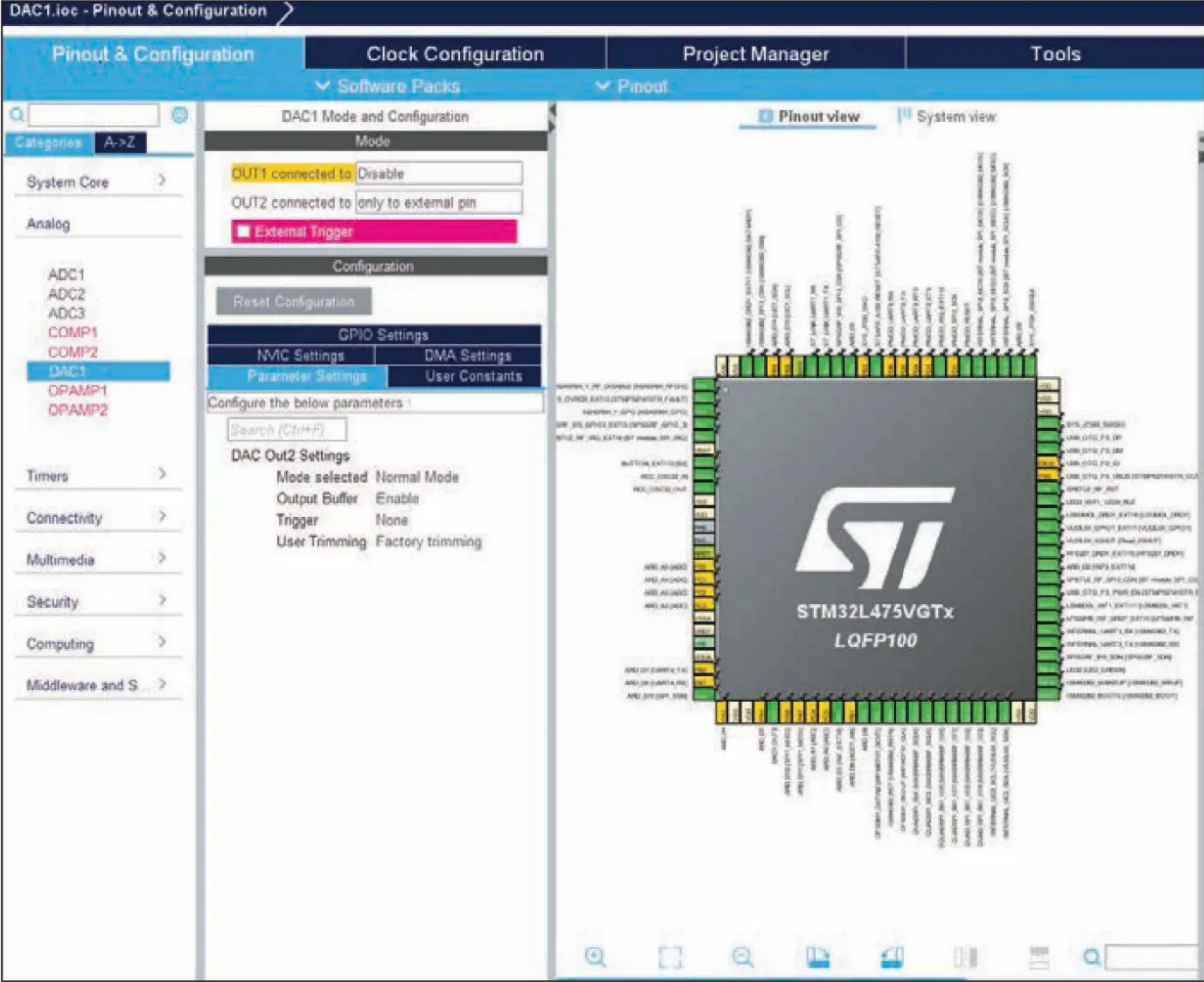


Fig.6: the Device Configuration Tool in STM32CubeIDE.

interfaces and general-purpose digital I/O (GPIO). Your code sets up which pins will be used for what, and most on-chip peripherals will have various options and modes of operation which have also to be configured in code.

There will also be some code required for general processor initialisation (eg, core clock speed and source). Importantly, when using a development board, the processor and its I/O will need to be configured to suit the board, even if not all the features are being used.

Coding the initialisation and configuration manually can be tedious and error-prone, but the Device Configuration Tool allows you to set up the processor, pin functions and peripheral options via its GUI menus. On saving the Configuration Tool settings, the software writes the details to a file with extension .ioc in the project folder and generates the code required to set up the microcontroller as specified.

Once you have initialised a project in STM32CubeIDE, you end up with a `main.c` file with the autogenerated code in it. You now have a working project that does nothing to use as a starting point for adding the functionality you need. Throughout the `main.c` file, there are numerous comments of the form:

```
/* USER CODE BEGIN name */
...
/* USER CODE END name */
```

The name indicates the nature of the code concerned. Places where the `name` is a number are used for writing more general code. You enter your code on

lines between a `BEGIN` comment and the corresponding (same `name`) `END` comment. The `BEGIN` and `END` comments must not be changed.

This ensures that the code will not be overwritten (deleted) when an automatic code generation process is rerun (eg, if the configuration is updated and saved using the Device Configuration Tool).

Software libraries

It is not just initial configuration that is potentially difficult and time consuming when developing code for microcontrollers. Other code is required to control the complex on-chip hardware as programs execute. Usually, this involves writing or reading from specific registers on the chip and requires detailed knowledge of the microcontroller device hardware.

As the hardware is fixed on a given device, the same or very similar code will be used in many projects, so it is relatively straightforward for the microcontroller manufacturer to provide a library of functions to facilitate straightforward use of the microcontroller.

These are referred to as APIs (application programming interfaces) – they provide an interface between the user's code and the microcontroller hardware.

Many microcontroller devices from a given manufacturer will provide the same or similar functions, but the details of coding to achieve this may vary (eg, different values in different registers). An API library can provide generic and easy-to-use functions that work on many devices.

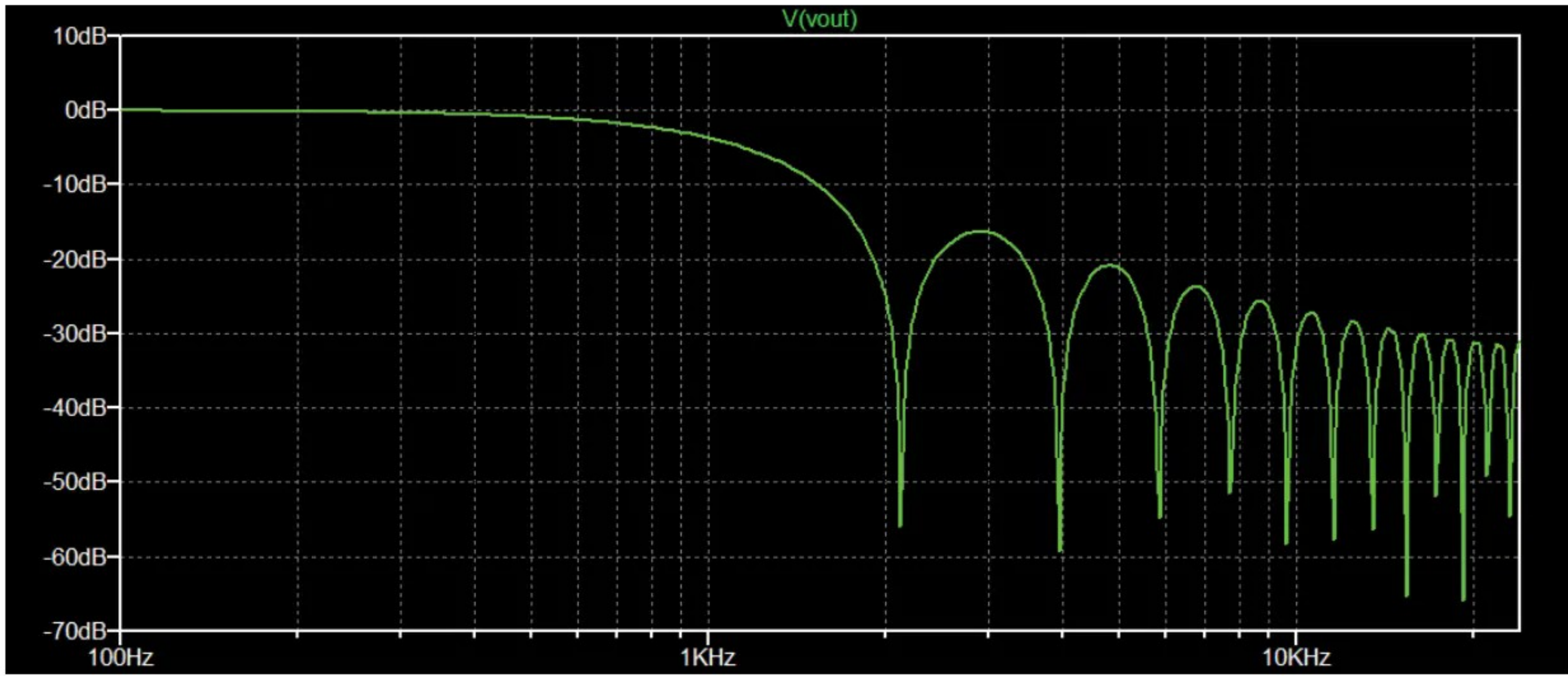


Fig.7: a frequency response simulation of our digital filter.

Of course, each device may also have more individual hardware features, accessed through APIs specific to that device. This set of generic and easy-to-use API functions is referred to as a hardware abstraction layer (HAL).

They ‘abstract’ in that they hide the specific details of the hardware, and ‘layer’ refers to the fact that they can be visualised as being in-between the user code and hardware. The term ‘driver’ may also be used to refer to software that facilitates interaction with hardware.

The HAL driver APIs make coding more productive, portable and less error prone. The relevant library is automatically available in projects created by STM32CubeIDE. For the STM32L475, documentation is provided in the STM32L4/L4+ HAL and low-layer drivers user manual (UM1884, see <https://pemag.au/link/ac5h>). Code generated by the Configuration Tool uses the HAL library.

Arm’s Common Microcontroller Software Interface Standard (CMSIS) DSP library can be used with STM32CubeIDE, but we will not use this to implement the example filter to keep the set-up simple. The filter maths will be directly implemented in the project, not hidden in a library. It’s more educational that way.

Making measurements

For an experimental implementation of a digital filter, we need to be able to supply a test signal at the input and measure the input and output signals so we can observe the effect the filter has on the signal. Ideally we want to be able to plot the frequency response

curve of the filter, similar to the plots from the LTspice simulations in previous articles (see Fig.7).

This requires signal generation and amplitude measurement over a suitable frequency range.

Some oscilloscopes with built-in signal generators can automatically plot frequency responses. If not, a signal generator and oscilloscope can be used to manually measure the gain at different frequencies. However, to see the sharp detail of frequency response, we need a lot of data points, which could be very time-consuming to gather manually.

A multimeter could also possibly be used to measure signal voltage, but many (particularly lower-cost models) only measure AC voltages over a very limited frequency range, so they would not be suitable. Devices that cover at least the 20Hz-20kHz range with accuracy are often termed ‘audio millivoltmeters’.

Fortunately, there is an alternative to purchasing expensive test equipment; it is possible to use the sound card on a PC or laptop as a signal generator/oscilloscope combination using free software tools, some of which will plot frequency responses automatically.

Sound cards have significant limitations as test equipment, but they can handle AC signals in the audio range at line level, which is more or less what is needed here. So they are a viable option that we will look at later.

System structure

Our example filter implementation will follow the structure of the generic DSP system in Fig.1. It is worth

considering exactly what we need and how the blocks in Fig.1 map to the development board, processor, and what additional circuitry we might be required.

We must decide which specific processor peripherals/pins to use (eg, which ADC if the processor has several), and if we are using a development board with peripheral circuits on the board, we need to check if they will influence our implementation.

The first block in Fig.1 is the anti-aliasing filter. We do not need this for measuring frequency responses with sinewave inputs that do not exceed the Nyquist frequency, as there is nothing for the filter to remove.

The ADC block is implemented by an ADC on the microcontroller. The sample-and-hold function is integrated into the on-chip ADC. The sampling/ADC conversion timing depends on the microcontroller hardware configuration, and software written to control this, which we will examine later.

The STM32L475 has three 12-bit 5Msps (megasamples per second) ADCs on-chip with 16 input channels (possible microcontroller pin connections) shared between them (see the datasheet at <https://pemag.au/link/ac5i>).

We need to check the development board schematic to see which microcontroller pins are available as ADC channels that are connected conveniently on the board and choose one to use.

For the board used here, ADC1 is most convenient because it can be fed from microcontroller pins PC0 to PC5, which are connected in turn to the six pins of the Arduino Uno header CN4 on the board (labelled ARD-A5-ADC to ARD-A0-ADC). No other circuitry is connected to these pins.

The relevant parts of the 11-page board schematic are shown in Fig.8 (this was extracted from the the user manual at <https://pemag.au/link/ac5j>).

ADC configuration

The Device Configuration Tool can be used to set which channel each ADC will be fed from. The default configuration of a new STM32CubeIDE project for the B-L475E-IOT01A board already has pins PC0 to PC5 configured as possible ADC1 inputs. All that is needed for basic use of the ADC is to enable the ADC1 channel that is connected to the microcontroller pin to be used.

We will connect ADC1 channel 1 (IN1) to pin PC0 (CN4 header pin A5).

To configure the ADC, click on Analog in the Categories list in the Configuration Tool to expand it, then click on ADC1 to select ADC1 for configuration. In the Mode part of the ADC1 Mode and

ARD.A5-ADC	15	PC0	PC8	65
ARD.A4-ADC	16	PC1	PC9	66
ARD.A3-ADC	17	PC2	PC10	78
ARD.A2-ADC	18	PC3	PC11	79
ARD.A1-ADC	33	PC4	PC12	80
ARD.A0-ADC	34	PC5	PC13/WKUP2	7
VL53L0X XSHUT	63	PC6	PC14-OSC32_IN	8
VL53L0X GPIO1 EXTI7	64	PC7	PC15-OSC32_OUT	9
STM32L475VGTx				

Fig.8: the ADC wiring on the B-L475E-IOT01A board from its user manual.

Configuration section, select Single-ended from the drop-down menu for IN1 (see Fig.9).

Similar procedures are used to configure all the microcontroller hardware used in a project. We will discuss other ADC configuration options later.

The digital processing part of Fig.1 is performed by code running on the microcontroller. The fundamental requirement for this example is to implement the FIR filter maths on the input signal.

The actual calculation is relatively straightforward – multiplying by coefficients and summing the results. However, the data must flow from the ADC to the calculation, and the results to the DAC in an efficient manner, with the sampling occurring at the correct rate. We will discuss the details of this later.

DAC configuration

The DAC is also internal to the microcontroller; the STM32L475 has a 12-bit DAC with two output channels. The settling time is 3µs to ±0.5 LSB, which would correspond to an update rate of around 300kHz. This is slower than the ADC, but more than enough for audio output.

The DAC has a sample-and-hold mode, which might seem confusing as sample-and-hold is usually associated with ADCs. It refers to using a capacitor to hold the DAC output voltage with the DAC circuitry powered down when the microcontroller is in a low-power mode. It is not needed here (normal mode is used).

As with the ADC, we need to look at the board schematic to see how the DAC is connected. The two output channels are available on pins PA4 and PA5. We will use DAC1’s Channel 2 via microcontroller pin PA5. The signal is labelled ARD.D13-SPI1_SCK/LED1 on the schematic.

The PA5 pin is connected to user LED1 on the development board (see Fig.10). The signal from the microcontroller is buffered by an op amp unity-gain amplifier (U21) so the LED will not load the DAC output. If the microcontroller is configured to use the DAC on pin PA5, the DAC will see a 210kΩ grounded load due to resistors R16 and R38.

The op amp and LED circuit can be isolated by desoldering solder bridge SB1 (on the underside of the board), but this is not necessary to use the DAC.

The ARD.D13-SPI1_SCK/LED1 signal is also connected to the Arduino Uno header CN1 pin (see Fig.11). This provides a convenient way to access the DAC output signal.

The default configuration of the PA5 pin is as SPI1_SCK (SPI serial bus #1’s

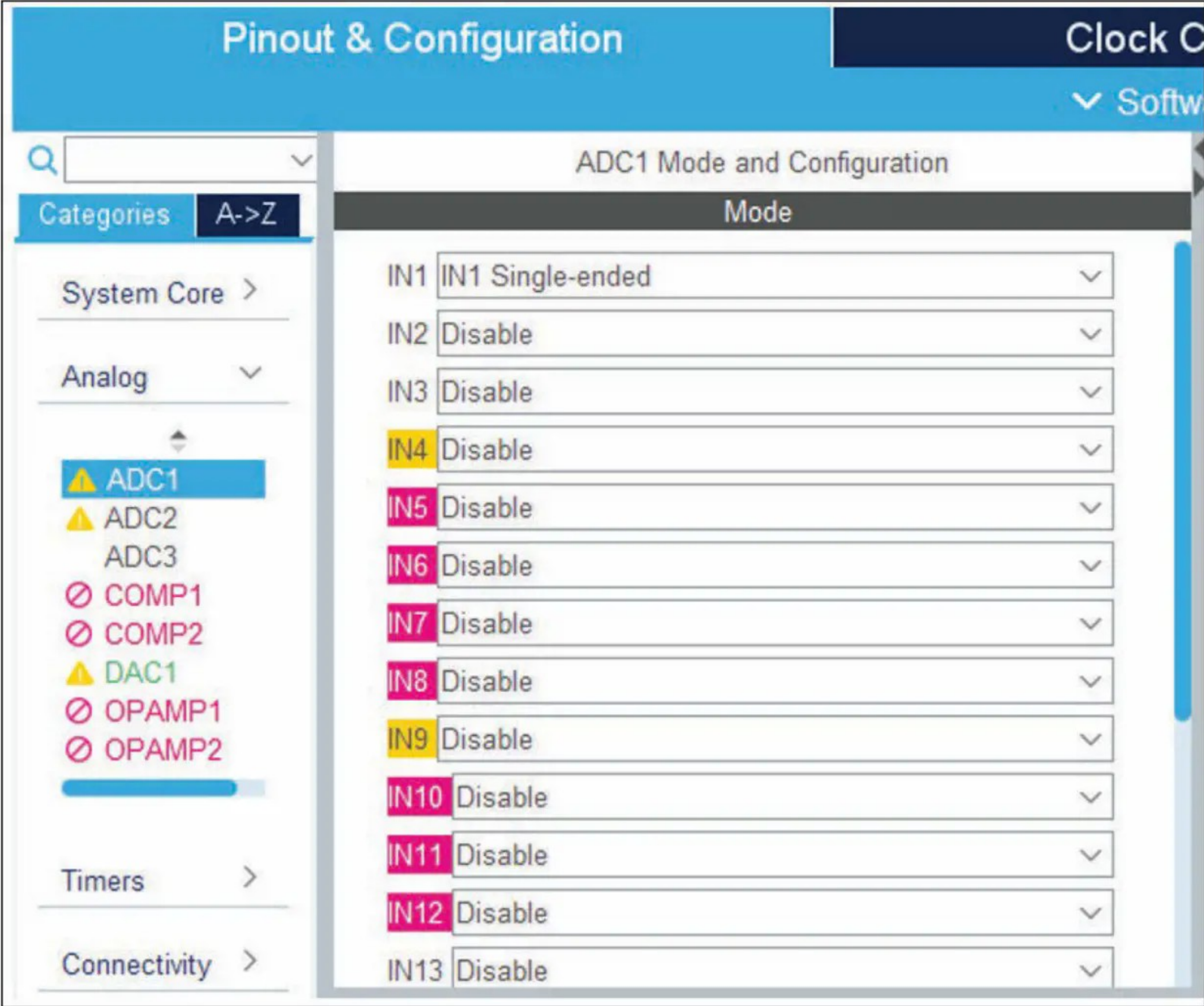


Fig.9: the ADC mode configuration we will be using.

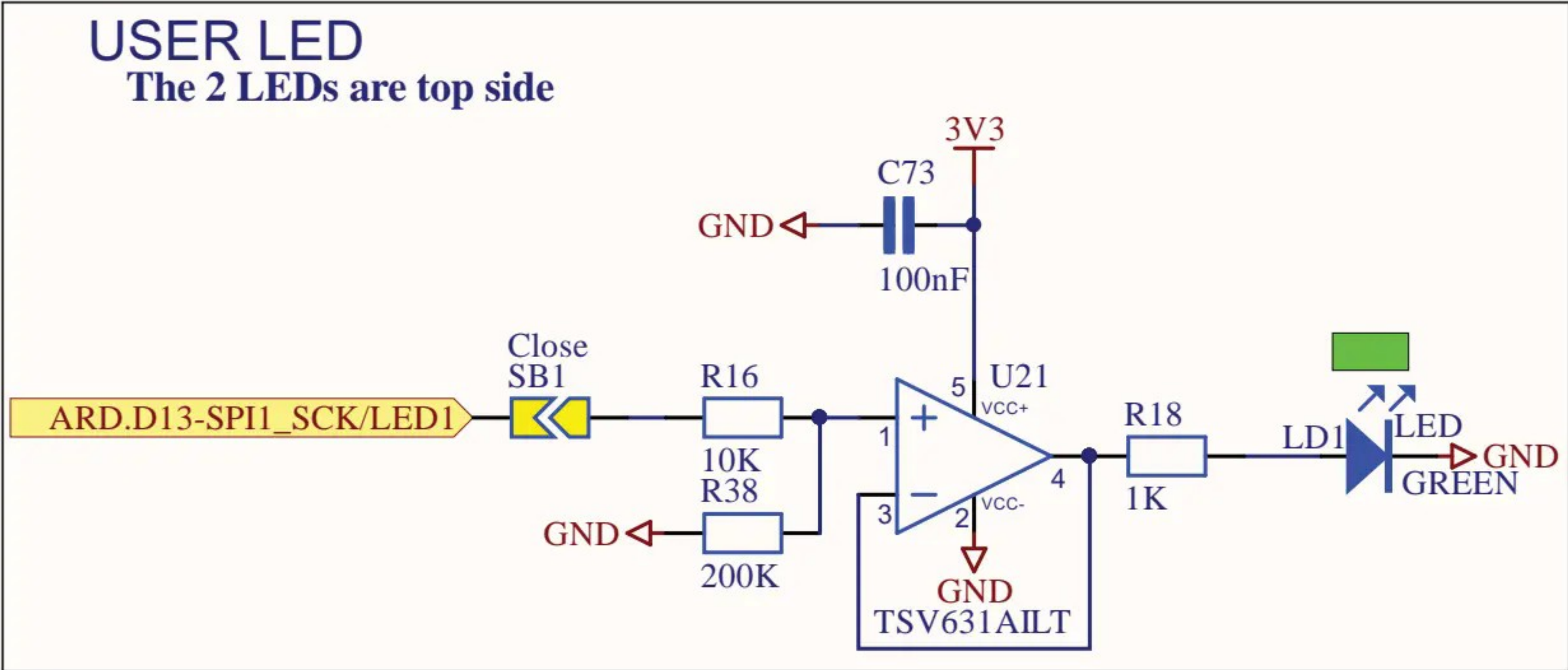


Fig.10: the LED circuitry connected to the PA5 pin on the B-L475E-IOT01A1 board.

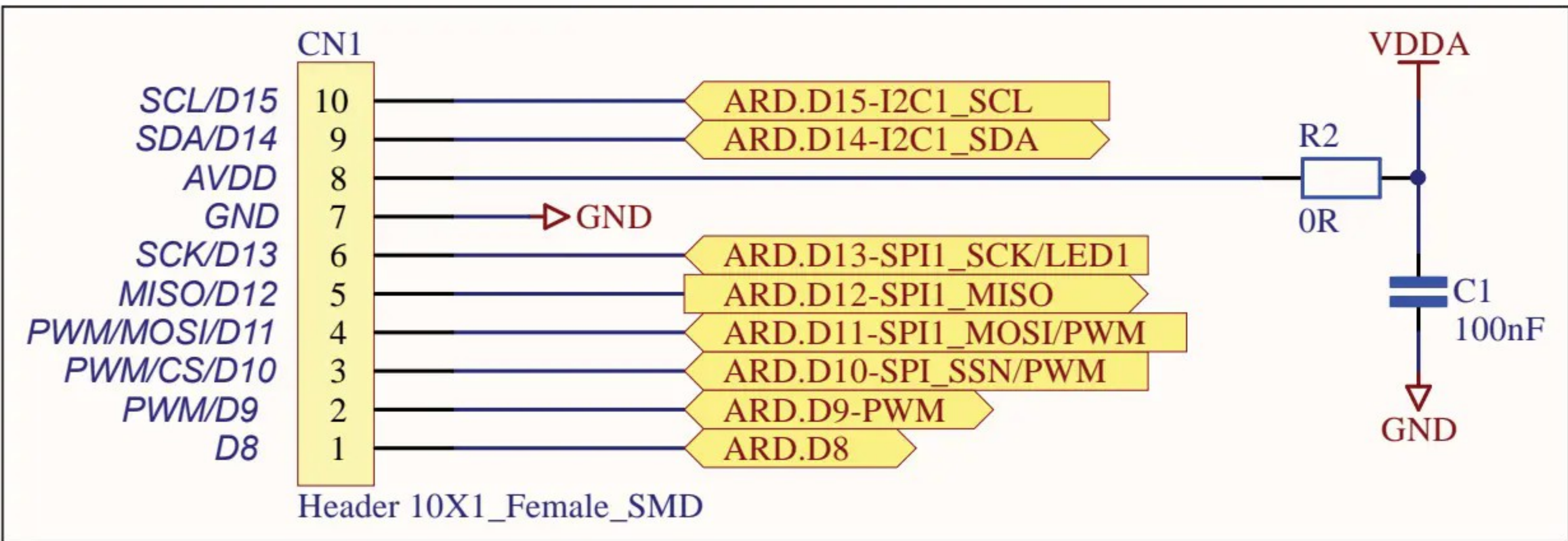


Fig.11: the DAC1 (PA5) connection to the Arduino header on the B-L475E-IOT01A1.

clock signal). To use the DAC, we change the PA5 pin configuration by clicking on it in the Device Configuration Tool and selecting the DAC1_OUT2 option from the pin menu (see Fig.12).

Like the ADC, it is necessary to configure the DAC. In the Device Configuration Tool, click on Analog in the Categories list and click on DAC1 to select it. In the Mode panel, change OUT2 from “Disable” to “only to external pin” – this will enable the DAC and route its output signal to the PA5 pin.

The configuration panel allows other options for the DAC to be set, which we will discuss in more detail later.

Pulses and power supplies

When exploring the operation of the FIR filter and subsystems, such as the DAC and ADC, it is useful to have a separate digital output to provide a time marker for when certain actions take place. For example, a pulse to logic 1 can be produced each time software attempts to update the DAC output.

To do this, we need to choose a pin to use as a general-purpose digital I/O and configure it as an output (GPIO_OUTPUT).

The PB2 pin is suitable for this is – it can be used as a GPIO and is only connected to the Arduino Uno header CN1 pin 1 on the development board. Therefore, it does not interact with any onboard circuitry. As pin PB2 is configured as a GPIO output by default, it is unnecessary to use the Configuration Tool to set this up.

The final block in Fig.1 is the reconstruction filter. We can observe the DAC output on an oscilloscope without using a filter, and if we use a reasonably high sampling rate compared with the signals, a simple RC low-pass filter may provide adequate filtering for basic measurements.

The B-L475E-IOT01A board can be powered via the ST-Link USB connection and provides +3.3V and +5V on the Arduino Uno connector.

The ADC input range and DAC output range are determined by their reference voltage inputs. On the B-L475E-IOT01A board, the microcontroller's VREF+ pin is connected to the 3.3V analog power supply (VDDA) and the VREF- pin is connected to ground (0V). This means the ADC and DAC ranges are both 0 to 3.3V.

When using the board, we have to make sure the signal input to the ADC does not go outside the 0V to 3.3V range. PE

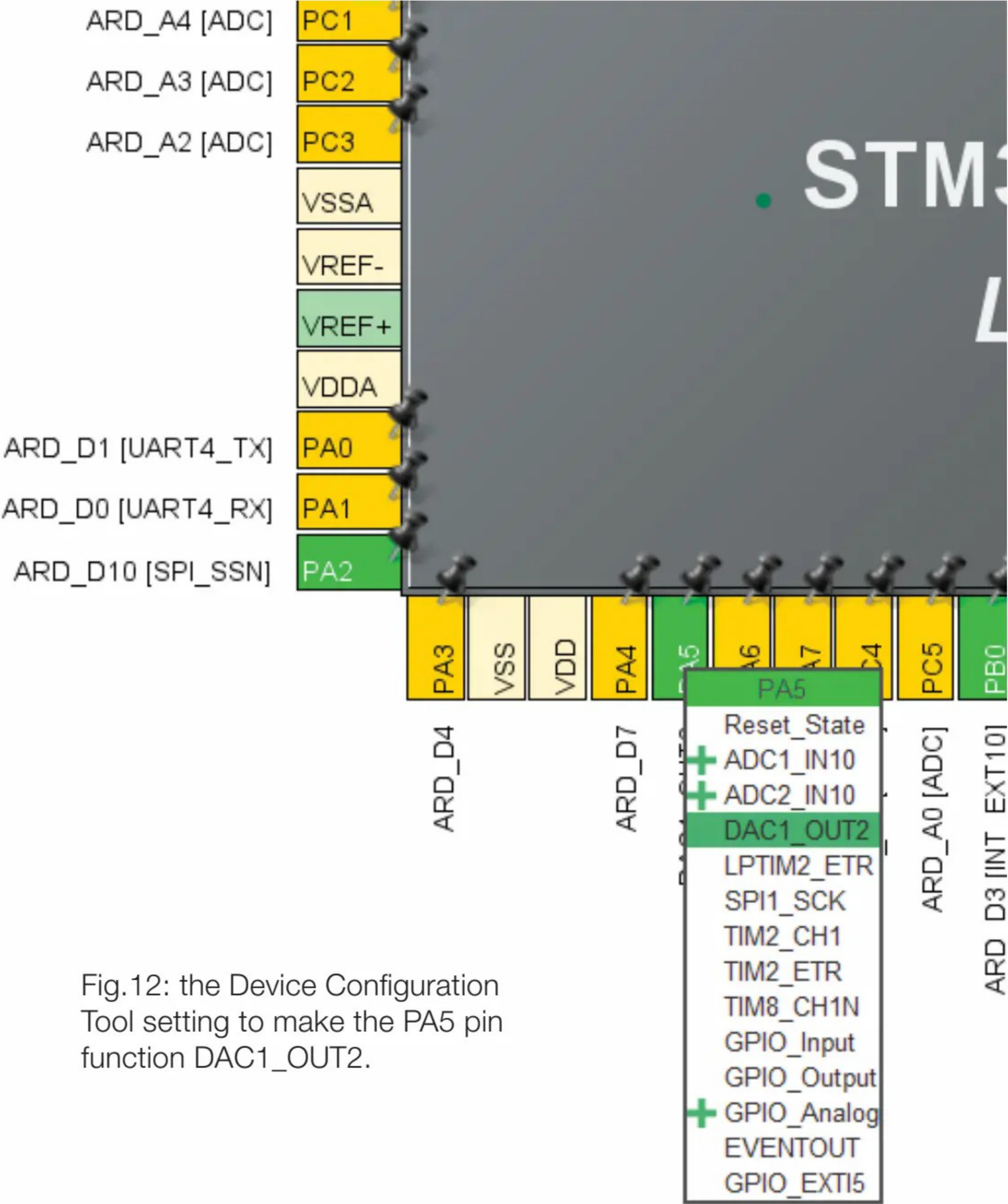


Fig.12: the Device Configuration Tool setting to make the PA5 pin function DAC1_OUT2.

JTAG Connector Plugs Directly into PCB!!

No Header! No Brainer!

Our patented range of Plug-of-Nails™ spring-pin cables plug directly into a tiny footprint of pads and locating holes in your PCB, eliminating the need for a mating header. Save Cost & Space on Every PCB!!

Solutions for: PIC . dsPIC . ARM . MSP430 . Atmel . Generic JTAG . Altera Xilinx . BDM . C2000 . SPY-BI-WIRE . SPI / IIC . Altium Mini-HDMI . & More

www.PlugOfNails.com

Tag-Connector footprints as small as 0.02 sq. inch (0.13 sq cm)

www.enigma-shop.com

Electronic Components for the Hobbyist!

Cable & Wire • Capacitors • Connectors • Filters
Kits & Modules • Hardware & Cooling
and much more...

READY TO BE REWARDED?

Join our loyalty scheme and **earn points** on every purchase you make which can be **used for discounts** in the future.

Read our reviews on **Trustpilot**

DOWNLOAD YOUR PE/EPE BACK ISSUES

COMPLETE-YEAR collections – all 12 issues for each year!
From just £5.95 each

VOL 28 – January to December 1999 – £5.95	VOL 41 – January to December 2012 – £14.95
VOL 29 – January to December 2000 – £6.95	VOL 42 – January to December 2013 – £15.95
VOL 30 – January to December 2001 – £7.95	VOL 43 – January to December 2014 – £16.95
VOL 31 – January to December 2002 – £7.95	VOL 44 – January to December 2015 – £17.95
VOL 32 – January to December 2003 – £7.95	VOL 45 – January to December 2016 – £19.95
VOL 33 – January to December 2004 – £7.95	VOL 46 – January to December 2017 – £22.95
VOL 34 – January to December 2005 – £8.95	VOL 47 – January to December 2018 – £27.95
VOL 35 – January to December 2006 – £8.95	VOL 48 – January to December 2019 – £27.95
VOL 36 – January to December 2007 – £8.95	VOL 49 – January to December 2020 – £29.95
VOL 37 – January to December 2008 – £8.95	VOL 50 – January to December 2021 – £32.95
VOL 38 – January to December 2009 – £10.95	VOL 51 – January to December 2022 – £34.95
VOL 39 – January to December 2010 – £11.95	VOL 52 – January to December 2023 – £34.95 – NEW!
VOL 40 – January to December 2011 – £12.95	VOL 53 – January to December 2024 – £34.95 – NEW!

FIVE-YEAR collections – 60 issues!
From just £14.95 each

PE2832: Jan 99 to Dec 03 – £14.95	PE3943: Jan 10 to Dec 14 – £33.95
PE2933: Jan 00 to Dec 04 – £14.95	PE4044: Jan 11 to Dec 15 – £33.95
PE3034: Jan 01 to Dec 05 – £14.95	PE4145: Jan 12 to Dec 16 – £35.95
PE3135: Jan 02 to Dec 06 – £14.95	PE4246: Jan 13 to Dec 17 – £37.95
PE3236: Jan 03 to Dec 07 – £14.95	PE4347: Jan 14 to Dec 18 – £38.95
PE3337: Jan 04 to Dec 08 – £14.95	PE4448: Jan 15 to Dec 19 – £39.95
PE3438: Jan 05 to Dec 09 – £19.95	PE4549: Jan 16 to Dec 20 – £42.95
PE3539: Jan 06 to Dec 10 – £25.95	PE4650: Jan 17 to Dec 21 – £44.95
PE3640: Jan 07 to Dec 11 – £25.95	PE4751: Jan 18 to Dec 22 – £44.95 – NEW!
PE3741: Jan 08 to Dec 12 – £25.95	PE4852: Jan 19 to Dec 23 – £44.95 – NEW!
PE3842: Jan 09 to Dec 13 – £33.95	

20 YEARS OF PE!
AMAZING BUNDLE OFFER
Every issue from 1999 – 2018
240 magazines for just
£69.95

PE2832: Jan 99 to Dec 03
PLUS!
PE3337: Jan 04 to Dec 08
PLUS!
PE3842: Jan 09 to Dec 13
PLUS!
PE3447: Jan 14 to Dec 18

Download your collections at: www.electronpublishing.com

Dual Mini LED Dice

This article is a blend of the old and the new. It's similar to a Dual LED Dice design from 1994 but has been updated to use more modern parts (still with discrete logic) and runs from a 3V coin cell. As we have used mostly SMDs (on the larger side), it will easily fit in your pocket.

Project by Nicholas Vinen



This small board 'rolls' two six-sided 'dice' each time you activate it, giving you a pair of random numbers in two different colours. It's small and light at just 60 × 28 × 15mm so it's convenient to use. You could even build two or three for games that require more than two dice to be rolled.

I am sure there are plenty of dice apps on smartphones these days, but there's something pleasing about a design based on old-fashioned discrete logic with Das Blinkenlights (in this case, 14 LEDs). I also think it's interesting to have such simple circuitry that does a useful job and only draws a few milliamps. It even switches itself off automatically, so the small coin cell should last a long time.

One thing I did to make it a little more interesting is add the option of triggering the dice roll with a vibration sensor. That way, you can shake it to roll! That part is optional, but it's pretty fun as you can just pick it up and quickly get some random numbers.

Coming up snake eyes

My original plan was to shamelessly copy borrow the circuit from a May

1994 project article in SILICON CHIP magazine by Darren Yates, change the parts to make it run from a lower voltage and redesign the PCB to be smaller. However, I quickly ran into a problem.

He had used four 4000-series logic ICs: two 4015 dual 4-bit shift registers and two 4093 quad schmitt-trigger NAND gates. The shift registers kept track of the state of the dice and also did some of the 'decoding' to drive the LEDs (more on that later). The NAND gates formed the oscillators to 'roll' the dice & performed some logic to always keep the shift registers in valid states.

There are direct equivalent 74-series logic chips to the 4093 NAND gates, such as the 74HC132, which would run from a 3V supply. However, I could not find any such equivalent of the 4015 dual 4-bit shift registers, at least, not at a reasonable price. There are 74-series shift registers but single eight-bit shift registers seem to be much more common/popular than dual 4-bit types.

So, while it might be possible to base a new circuit off the old one, it would make building it quite expensive, which I thought was against the

spirit of the project. I wanted to have cheap kits, under \$20 each, to make it a fun device that you can build several of if you are so inclined.

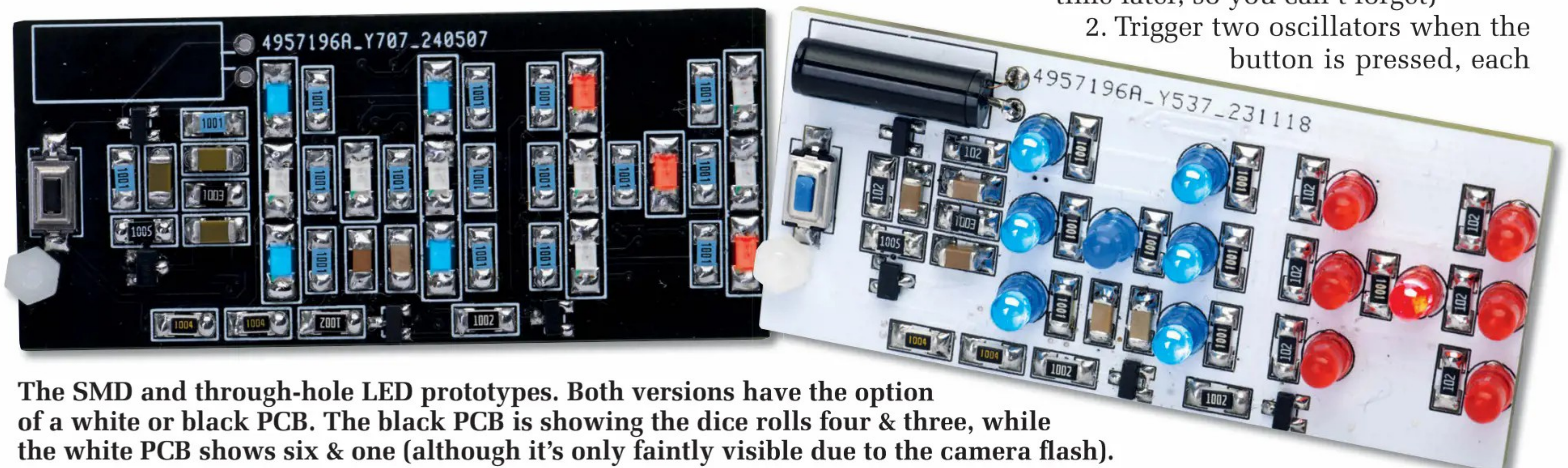
So, back to the drawing board then, to come up with an equivalent circuit using more modern (readily available and inexpensive) parts. While I was revising the design, I thought I would see if I could come up with a way to do it with fewer chips. Spoiler: my design uses just three to do the same job (with two spare logic gates!). But before we get to that, let's look at the problem I had to solve.

Of course, I considered using a PIC to do this, but what's the fun in that? The software would be trivial and the resulting board would be tiny and pretty cheap to build, but it would be a 'black box'. Keeping with the discrete logic means that anyone can understand how the circuit works.

Rolling the bones

There are basically four things a battery-powered circuit needs to do to emulate rolling two dice:

1. Switch on when the button is pressed (and switch itself off some time later, so you can't forget)
2. Trigger two oscillators when the button is pressed, each



The SMD and through-hole LED prototypes. Both versions have the option of a white or black PCB. The black PCB is showing the dice rolls four & three, while the white PCB shows six & one (although it's only faintly visible due to the camera flash).

of which increments the number on a die, going through the sequence 1, 2, 3, 4, 5, 6, 1... with decreasing frequency, so it eventually stops on two numbers.

3. Keep track of what number each die is currently showing.

4. Convert that number (1-6) into a pattern of LEDs akin to the dots on the face of a traditional die.

For #1, I decided on a trick we've used a few times in previous projects: a Mosfet with a capacitor and parallel resistor between gate and source, plus a second resistor (and in this case, diode) to pull the gate up when the button is pressed. The RC time constant of the first two components sets the maximum time the Mosfet will remain on, powering the circuit, before switching itself off.

The advantage of this approach is its simplicity and low cost, requiring just one small Mosfet (as the circuit's current requirements are low) and a few passives. The disadvantage is that it switches off by slowly lowering the supply voltage. That means the dice LEDs fade out rather than just switching off, but I can't see the harm in that. That gives you a bit of warning that it's going to switch off!

For #2, I copied the design from the May 1994 circuit, where the same button that switches the unit on also charges up a pair of capacitors that control two oscillators using schmitt-trigger NAND gates as inverters. The voltage on the capacitor affects the oscillator rate, so they slow down and then stop when you release the button.

I made one change here; the original circuit used two capacitors of identical values and relied on the fact that no two capacitors will be exactly the same value to cause the oscillators to desynchronise, so you don't get the same numbers on each die.

I found that did not work too well – in one test, I had 20 rolls in a row where both dice gave the same number! I think this was because the oscillators were close enough that feedback through the power supply was locking them together. Using two different values for the capacitors fixed that, and I think it's more pleasing that the 'dice' stop at different times; just like real ones.

#3 and #4 are where the designs really differ, and this is where I was able to save one IC. Unlike the 1994 design, where state-keeping and decod-

ing were somewhat muddled together, my design keeps them mostly separate.

To keep track of the state of each die, instead of using shift registers, I am using the registers in a dual 4-bit counter IC, the common 74HC393. Normally each counter will go from 0 to 15 and then back to 0, repeating forever, with the counter incrementing on each clock pulse. However, we want it to roll back to zero after five, so it has six discrete states.

We achieve that by creating a 'crude' AND gate for each counter out of a dual common anode schottky diode. We connect the cathodes to the O1 and O2 outputs, the anode to the CLR input and pull the CLR input up with a resistor.

Fig.1 shows how the O1 and O2 outputs both go high for the first time when the counter reaches 6. It is at this point that the diode stops conducting, allowing the pull-up resistor to assert the clear input, causing the counter to reset to zero. When it resets, O1 and O2 go low, so clear is immediately deasserted. This causes the counter to go 0, 1, 2, 3, 4, 5, 0, 1, 2, 3...

LED driving

So we have our die states and we can roll them, but how do we drive the LEDs? I spent a couple of hours pondering how to convert the O0-O2 outputs of each counter to the six required LED states that are shown in Fig.1, trying to find the absolute minimum of low-cost logic to do it.

The logic required can be minimised by driving some LEDs from one end, with others driven at both ends. By driving the LED from both ends, we effectively get a 'free gate', because it will only light in one of the four possible states of a pair of digital outputs.

It will light with the anode pulled high and the cathode low. In two other states (low/low and high/high), there is no voltage across the LED. In the fourth, it is reverse-biased and will not conduct (at least, not with the meagre 3V we are applying).

Complicating things a bit is the fact that our counter doesn't go from 1 to 6, but from 0 to 5. I considered that we don't necessarily need the numbers to come in order; as long as all are present and equally likely. However, I figured out a way for them to occur in order, so I kept it that way.

The problem with mapping counter values of 0-5 to die face numbers of 1-6

is that the O1 and O2 outputs change on the counter transitions from 1 to 2, 3 to 4 and 5 to 0 on the counter. That would correspond to die face values of 2 & 3, 4 & 5 and 6 & 1, when those are actually the most similar states (two of them differing only by the state of the middle LED).

Instead, I decided that a counter value of zero should show six on the die face, with the other five values (1-5) mapping to those same values on the die face. That makes decoding much easier, but keeps the numbers in sequence (6, 1, 2, 3, 4, 5, 6, 1, 2...).

Circuit details

Having decided on that, we can immediately sort out the central LED. It is always lit for odd numbers but never for even numbers. As shown in Fig.1, O0 is always high for odd die case numbers and low for even ones, so we just need to connect the O0 output to the middle LED's anode (via a resistor) and connect its cathode to ground, as shown in Fig.2, and it will light at the right times.

Next, let's consider the two diagonal LEDs that will light initially to show two, then three, remaining on for four, five and six. We could chose either diagonal pair but I have opted for LED2 (upper left) and LED3 (lower right), as per Fig.1. The only die face number

O0	O1	O2	Count	Die face
0	0	0	0	
1	0	0	1	
0	1	0	2	
1	1	0	3	
0	0	1	4	
1	0	1	5	
0	1	1	6	

Fig.1: how the three binary counter outputs O0-O2 correspond to the counter value and die faces.

Constructional Project

where they are off is one; they are on for the five remaining possibilities.

The logic required to detect a one from the O0-O2 outputs is O0 AND NOT (O1 OR O2), which gives 1 for a die face of one and 0 for everything else. That can be rewritten as O0 AND (O1 NOR O2). NAND ICs are more common than AND, but that's OK because using one instead just inverts the result, meaning we get a result of 0 for a die face of one from IC2b/IC1c.

We therefore connect this gate output (pin 8 of IC1c) to the anodes of LED2 & LED3, connect their cathodes to ground, and they will light for any die face state but one.

So far, besides the counter IC, we just need two NOR gates and two NAND

gates for both dice. Two-input logic ICs usually have four gates each, so with one NOR and one NAND IC, we have two of each gate type left. We want to use two NAND gates for our oscillators, leaving us with just two NOR gates. Is that enough to drive the remaining four LEDs?

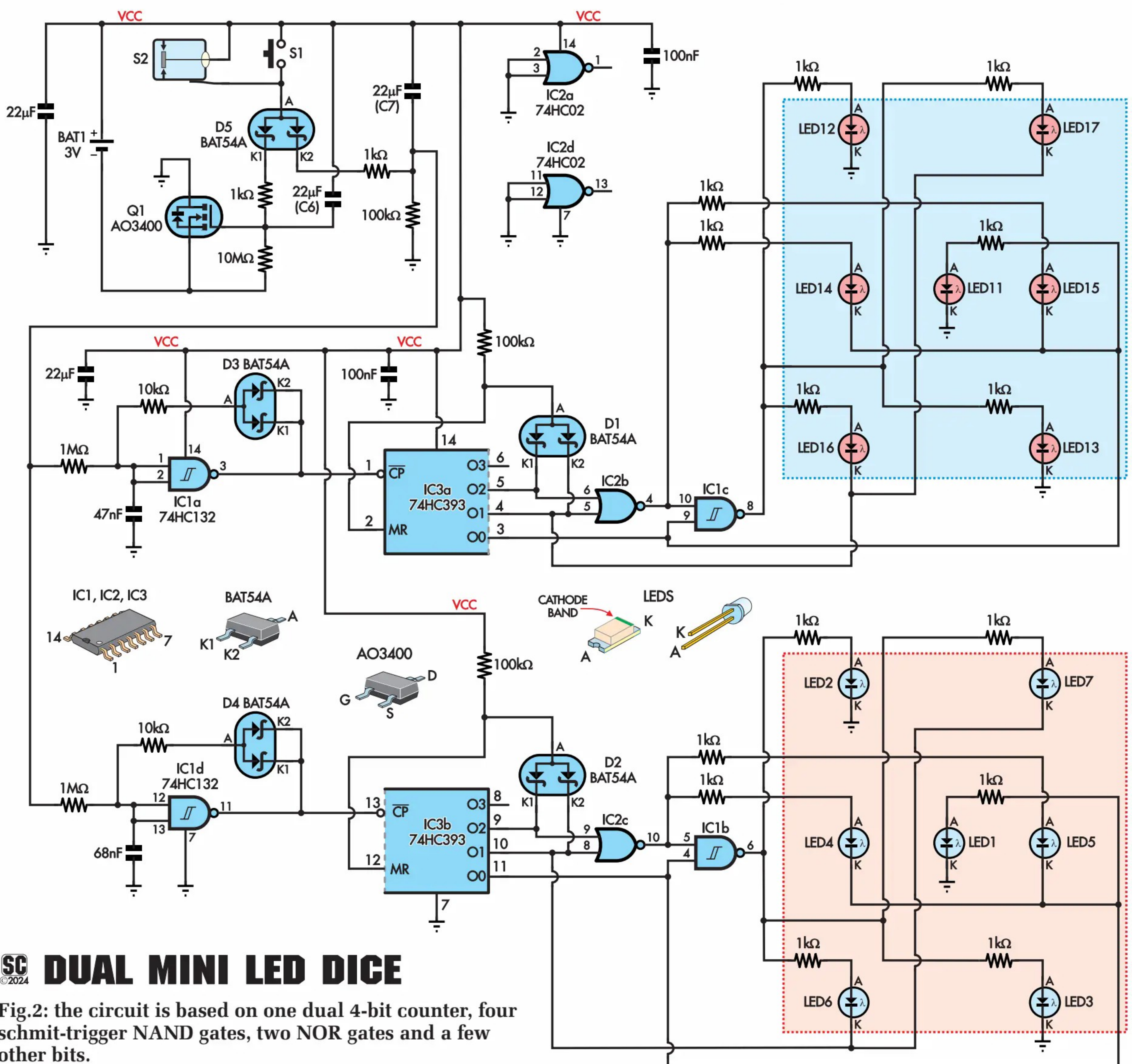
Actually, we don't need any more logic gates; we've already performed all the logic we need! The other two diagonal LEDs, LED6 & LED7, need to light for die face states of four, five and six. That's the same set of states as for LED2 & LED3, except the ones where the O1 output is high (two and three).

Therefore, all we need to do is connect the anodes of LED6 & LED7 to the same point as LED2 and LED3

and connect their cathodes to the O1 output. LED6 and LED7 will therefore light when LED2 and LED3 are, except when the O1 output is high. In that case, both ends of LED6 & LED7 will be at the same voltage. Therefore, LED6 and LED7 are off for values of 1, 2 & 3 and on for 4-6.

Finally, we have the middle LEDs on either side, LED4 & LED5. They only come on to show six, when all three digital outputs of the counter, O0-O2, are low. We already have a NOR gate (IC2b) combining outputs O1 and O2; its output will be high only for two die face values, one and six. So all we need to do is eliminate one.

So we connect the NOR gate output (pin 4 of IC2b) to the anodes of LED4



SC 2024 DUAL MINI LED DICE

Fig.2: the circuit is based on one dual 4-bit counter, four schmitt-trigger NAND gates, two NOR gates and a few other bits.

Dual Mini LED Dice

& LED5, and join their cathodes to the O0 output. They will only light when the anode is high (states one & six) and cathode is low (states two, four and six). Therefore, they only light up for six.

That's it – all LEDs are lit at the appropriate times, and we have two NOR gates to spare! I couldn't think of anything useful to do with them; I suppose they could have been used to buffer some LEDs, so the NAND gate didn't need to drive so many, but I found it easier to leave them unused and tie their inputs to GND.

Power supply and oscillators

22 μ F capacitor C6 is usually charged up to the full cell voltage, so Q1's gate-source voltage is 0V and it remains off. The circuit's ground is therefore disconnected from the bottom end of the cell, and the circuit is not powered. In case of any leakage, C6 is kept charged by the 10M Ω resistor between Q1's gate and source terminals.

When the contacts of S1 (tactile push-button) or S2 (vibration switch) close, current can flow from the positive terminal of the 3V cell via the two schottky diodes in D5 to two places. One of those current paths flows through a 1k Ω resistor to discharge C6, raising Q1's gate voltage to around 3V and switching it (and thus the rest of the circuit) on.

Once the switch is released, the 10M Ω resistor slowly recharges C6, eventually switching Q1 and the rest of the circuit off after about a minute.

22 μ F capacitor C7 is discharged at the same time, via a second 1k Ω resistor, but this one charges more quickly, via a 100k Ω resistor to ground. This produces the voltage that varies the oscillator speed from fast to slow, then stopped, to simulate the dice roll.

That voltage starts high when the switch is pressed and then drops. It is applied to the inputs of both schmitt-trigger inverters (IC1a and IC1d) via 1M Ω resistors, charging up the 47nF & 68nF capacitors from those points to ground. Once those capacitors charge to a certain point, the output of the inverter goes low, discharging the capacitor quickly via D3 or D4. The cycle then repeats.

We only use parallel diodes for D3 and D4 so that we can use identical diode parts throughout the circuit, making component sourcing and construction easier.

As the 'dice roll' voltage drops, the charge current through 1M Ω resistor drops, so it takes more and more time for the oscillator cycle to complete. When the voltage from C7 drops below the negative-going threshold of schmitt-trigger inverters IC1a and IC1d, they can no longer oscillate, and the dice display remains static until a switch contact closes or the unit powers down.

The reason we have both C6 and C7 is that we want C6 to charge more slowly, so the unit stays on for a while, but C7 charges fast so the dice roll completes within a couple of seconds.

There is a 22 μ F capacitor across the coin cell to improve its surge current capability, plus a 22 μ F bypass capacitor for IC1 and 100nF bypasses for IC2 and IC3. A high-value bypass capacitor is used for IC1 because we don't want voltage variations due to different LEDs lighting to affect the oscillators too much, or that could bias the dice rolls (increasing the chance of them stopping on certain numbers).

The different value oscillator capacitors (47nF & 68nF) ensure the oscillators run at different rates, so there is no relationship between the number shown on the two dice.

LED colours

You could use the same colour of LED for both die faces but we think it's helpful to have them be different colours. For example, if two people need to roll one die, you can assign them each a colour and roll them together. Still, it's up to you.

We chose blue and red because they both have a high efficiency and give similar brightness with 1k Ω current-limiting resistors running from a 3V supply. The red LEDs do draw a little more current, as they have a lower forward voltage, but both are pretty economical on power. The blue LEDs are quite bright at about 0.5mA while the red LEDs are similar at around 1.2mA.

We tried green LEDs and they barely lit up with the 1k Ω series resistors running from 3V. We considered lower resistor values but that would put quite a strain on the button cell. Another colour that could work well is white. Yellow or amber LEDs might work well if they are high-efficiency types.

A note on vibration sensors

One of the biggest challenges during

the development of this project was finding vibration sensors that actually worked! One of the most common such devices is the SW-18010P, which we have used before. However, it turns out that there a lot of dud/fake/counterfeit/badly made devices around sold under the name SW-18010P. So you have to make sure you get them from a reputable supplier.

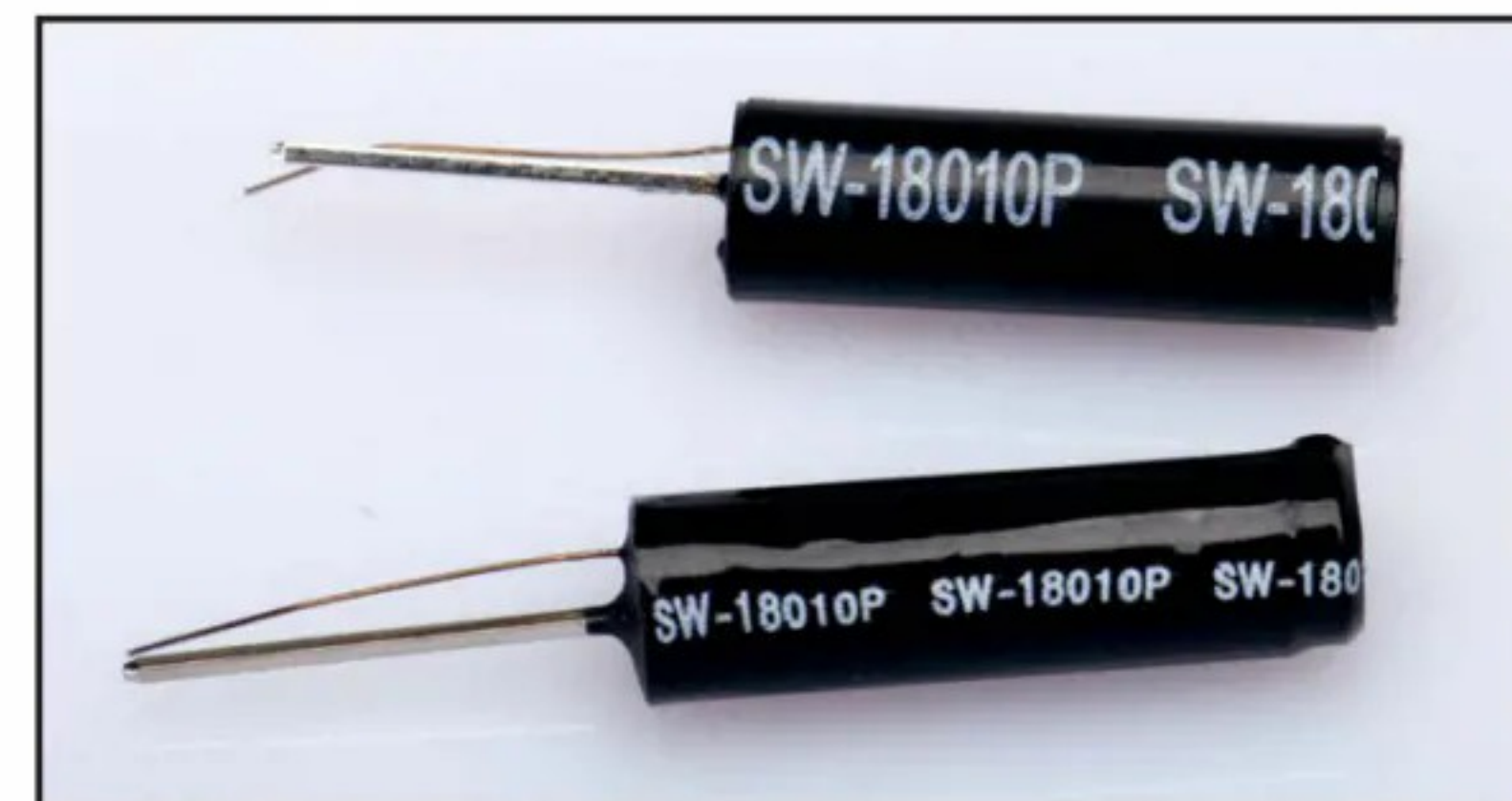
The first lot of SW-18010Ps we got were complete duds, despite getting them from a supplier who had sent us good parts previously. They have tiny writing on the black body, as shown below. While they showed some signs of life, you had to shake them at Earthquake Magnitude 10 level to get any sort of switch closure and it seemed very inconsistent. So in the bin they went.

Thinking that maybe the SW-18010P was not a good part to use, we looked for alternatives and found several likely ones: the SW-200D, SW-420D and SW-520D, all described as "highly sensitive vibration switches". We duly purchased some of each, and were shocked upon receiving them to find that they were all tilt sensors, not vibration sensors!

It's easy to tell that because you can hear a ball rolling inside them when you tip them, and they have a high resistance in some orientations and a low resistance in others, even when static. So they clearly were not suitable.

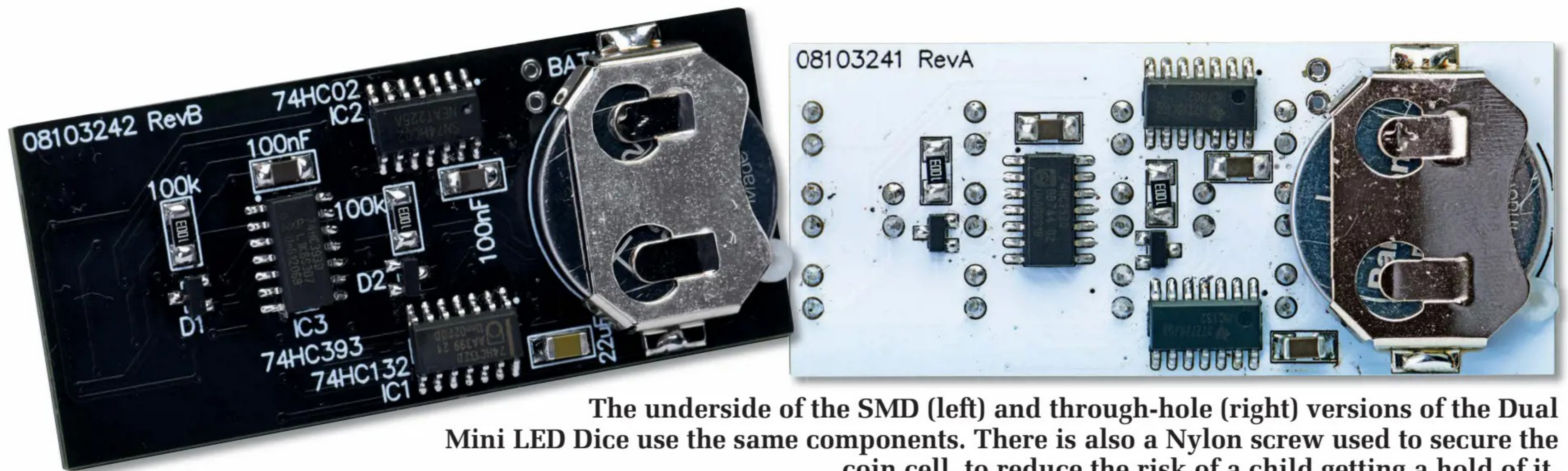
Finally, we found a seller online who actually supplied us with SW-18010P sensors that worked. As you can see in the photo, they have a slightly lighter body and larger writing. DigiKey also sells the AdaFruit version of this part (Cat 1528-2158-ND) which would be a good option if you need to buy it yourself.

Still, our kits will come with parts that we've checked and found to be working, so if you build this from a kit,



Two different SW-18010P vibration sensors we purchased. We found that the ones with smaller writing on the side were highly unreliable! The ones shown at the top work much better.

Constructional Project



The underside of the SMD (left) and through-hole (right) versions of the Dual Mini LED Dice use the same components. There is also a Nylon screw used to secure the coin cell, to reduce the risk of a child getting a hold of it.

you shouldn't have to worry too much about the sensor being functional. By the way, the less-sensitive SW-18015 and SW-18020 devices are probably no good because even the SW-18010P is barely sensitive enough (you have to give it a pretty firm shake to activate it).

While the vibration sensor makes it a very fun device to use, it is a bit of a gimmick. Even though you have to shake it fairly hard to get a good roll, accidental triggering is still a problem. For example, if you transport it in a car, it will roll the dice if you go over a pothole or big bump in the road. If you keep it in a pocket, it could be triggered while you walk, wearing down the battery.

If you're playing a game and depend on a good roll, we suggest you use the pushbutton to roll the dice as it seems to give better randomisation. Still, as long as you make sure you give it a good shake, it seems to work well enough, and it certainly will wake it up from sleep reliably.

Construction

Despite the design being mostly SMD based, we've chosen to use 3mm through-hole LEDs as we think they look more like the coloured dimples on a die face and, as their lenses project above the tops of the SMDs, they stop the other components on the board from detracting from the LED display.

We have produced an alternative PCB (coded 08103242) that uses SMA/M3216/1206 (imperial) sized SMD LEDs instead, for any constructors who might prefer the slimmer result. We could have designed a PCB to accept both but then we think it wouldn't have looked as good when using the 3mm through-hole LEDs.

Both PCBs measure 59.5 × 26mm and the overlay diagrams are shown in Figs.3 & 4. Components mount on both sides of the PCB. The top side mainly has the LEDs and their current-

limiting resistors, while all the ICs and the battery are on the other side. Once assembled, the whole thing can be encapsulated in a length of clear heatshrink tubing for protection.

We recommend that you start by mounting all the SMDs on the side of the PCB with the LEDs (the 'front'). The resistors will be labelled with codes like those shown in the parts list; you may need a magnifier to see them. The capacitors will not be labelled so don't get them mixed up once you remove them from their packages.

There are various ways to solder these components but the way we assembled the prototype was to put a little solder on one pad then, holding the part with tweezers, slide it into place while heating that solder. We removed the iron and let it solidify once the part was centred on its pads. We then checked its alignment and, if it was off, reheated the solder and gently repositioned the part with tweezers.

Once it was nicely centred and flat on the board, we soldered the opposite pad, ensuring we added enough solder for it to flow onto and adhere to both the pad and part. We then waited for that to solidify, added a tiny bit of flux paste to the initial joint and heated it with the iron tip to reflow it. Repeat until all the passives are in place on the top side.

Next, mount Mosfet Q1 (SOT-23) towards lower left. Use a similar technique but this time there are three pins to solder. Follow with the other SOT-23 package devices on the top side, diodes D3 through D5.

If you are building the board with SMD LEDs, fit them next. Don't get the different colours mixed up or it will look odd; use all the same colour LEDs for each die face. Ensure the cathodes are orientated as shown for LED1 in Fig.4. You can check this with a DMM set on diode test mode. Carefully touch the probes to the LED leads. When it lights up, the red probe is on the anode and the black probe on the cathode.

Now is a good time to clean any flux residue off this side of the board with isopropyl alcohol, methylated spirits or (ideally) a specialised flux cleaning formula. After that, flip the board over.

Parts on the other side

The three ICs mount on this side. All three are in similar 14-pin SOIC packages, so don't get them mixed up, and make very sure that you identify pin 1 and locate it as shown on the underside overlay. It's difficult to remove and refit an SMD IC unless you have a hot air station!

Use a similar technique as before, tacking one pin in place and checking that all the pins are aligned over their pads before soldering the other corner pins, then the remainder. You can add a little flux paste along both rows of pins and drag solder them, or just touch a soldering iron loaded with a little solder to each pin and the flux should draw it onto the pin and pad.

Don't be too concerned if you accidentally bridge two pins. Once all pins are soldered, check for bridges and, if you find any, add more flux paste to those pins and use a piece of



Both versions of the LED Dice (shown at actual size) can be covered with heatshrink for protection. You can then use it via the pushbutton, or by shaking it (if you have mounted the vibration sensor).

Dual Mini LED Dice

solder-wicking braid to draw off the excess solder. Once all three ICs have been soldered, clean off the flux residue and check that all the solder joints are good with a magnifier, and verify there are no bridges.

You can then fit the two remaining diodes on this side, then the three 100nF capacitors and two 100kΩ resistors. Clean off any new flux residue, then flip the board back over and solder the tactile pushbutton in place. Try to get it straight so it looks neat.

If using the through-hole LEDs, now is a good time to solder them in. With the board right-side up (the side that the LEDs sit on), the anodes (longer leads) all go towards the top, and the flat side of the lenses to the bottom. Insert each LED fully, then solder and trim the leads once you are sure it is sitting flat on the PCB.

Return to the underside of the board and tin one of the rectangular cell holder pads near the edge. Add a smear of flux paste onto both of those rectangular pads.

Rest the cell holder in place and make sure the entry side is facing the edge of the board (if you're unsure of the correct orientation, consult our photos). Add a bit more flux paste on top of the two tabs that rest on the PCB. Once you're sure it's lined up correctly, gently press it down and feed solder onto one of the tabs. You may need to turn your iron up due to the thermal mass of the metal holder.

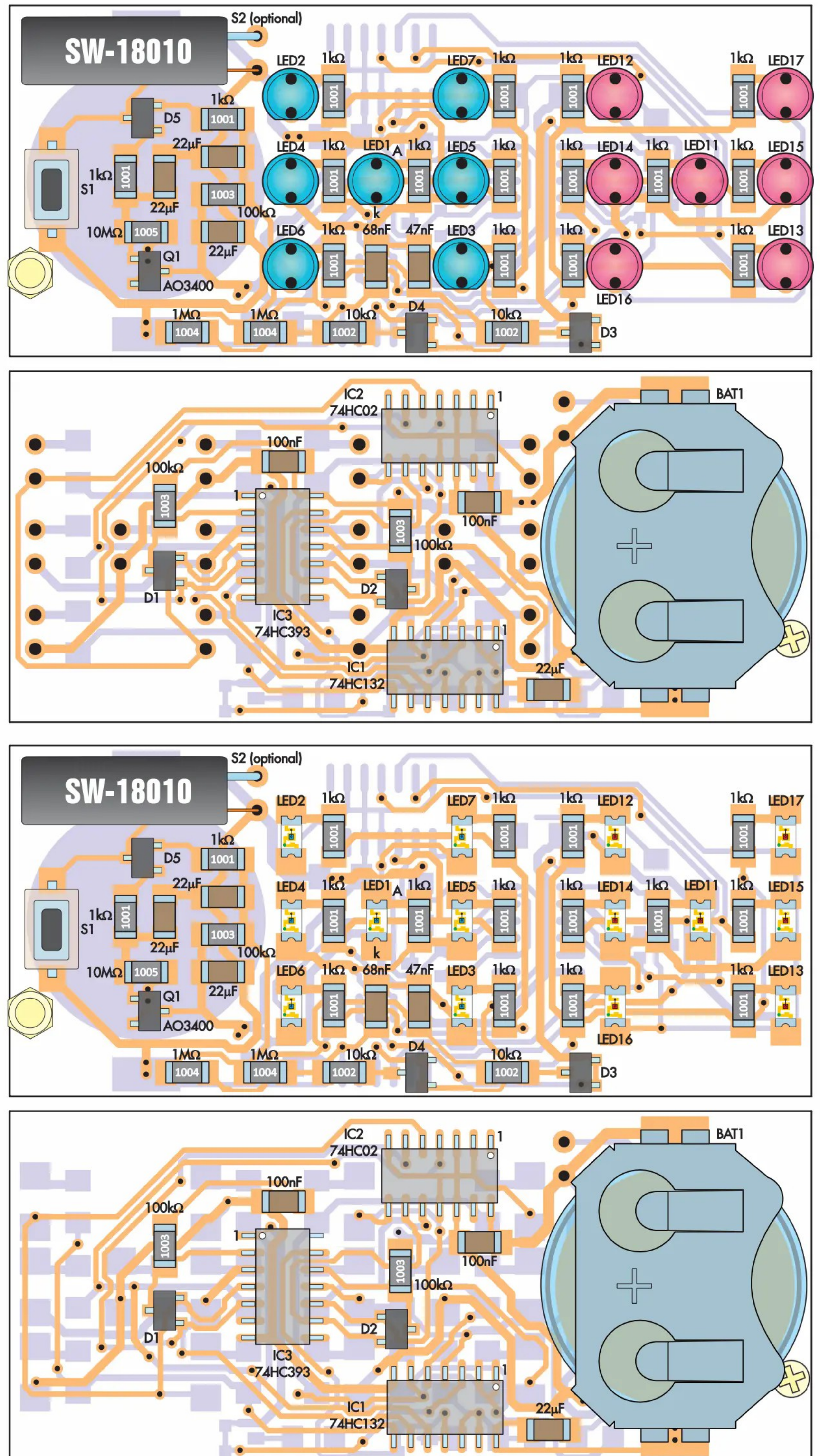
If you will be using the vibration switch, leave it off for now as it's easier to test the circuit without it.

Testing

If you have a current-limited bench supply, you can set it to 3V/50mA and connect it using clip leads. Clamp the red alligator clip to the metal shell of the cell holder but make sure it isn't touching any other components or tracks on the PCB. Carefully clip the black one to the edge of the PCB near the cell holder so it contacts the round pad under the holder but nothing else. Switch the supply on.

If you don't have that, you can just use a lithium coin cell. They can't deliver a lot of current and it's easy to temporarily slip one into the side of the cell holder, making enough contact to power the circuit but allowing you to quickly pull it out if something seems wrong.

Note that if you use a coin cell, the



Figs.3 & 4: we didn't find the coin cell shorted the adjacent LED leads, but make sure you trim them close to the PCB. For the SMD version, the LED cathodes all go towards the bottom of the PCB.

circuit will take a little while (probably 60s) to settle. The LEDs may be dim at first but should get brighter, assuming you are using a fresh cell.

When power is applied, you should

see the LEDs immediately light up and the dice roll. If that doesn't happen, or the circuit draws more than 20mA, switch it off check for incorrectly placed or soldered components. A common

Parts List – Mini LED Dice

- 1 double-sided PCB coded 08103241, 59.5 × 26mm •
- 1 SMD 20mm coin cell holder (BAT1)
- 1 CR2032 lithium coin cell
- 1 2-pin SMD tactile pushbutton (S1)
- 1 SW-18010P vibration sensing switch (S2) (optional)
[AliExpress [1005003297304614](https://www.aliexpress.com/item/1005003297304614)]
- 1 75mm length of 30-40mm diameter clear heatshrink tubing
- 1 M2 × 6mm nylon panhead machine screw and matching nylon hex nut

Semiconductors

- 1 74HC132 schmitt-trigger quad 2-input NAND gate CMOS IC, SOIC-14 (IC1)
- 1 74HC02 quad 2-input NOR gate CMOS IC, SOIC-14 (IC2)
- 1 74HC393 dual 4-bit binary counter CMOS IC, SOIC-14 (IC3)
- 1 AO3400 30V 5.8A N-channel logic-level Mosfet or equivalent, SOT-23 (Q1)
- 7 blue 3mm high-brightness diffused lens LEDs (LED1-LED7) •
- 7 red 3mm high-brightness diffused lens LEDs (LED11-LED17) •
- 5 BAT54A dual common-anode schottky diodes, SOT-23 (D1-D5)

Capacitors (all SMD M3216/1206 size 50V X7R unless noted)

- 4 22µF 6.3V 2 100nF 1 68nF 1 47nF

Resistors (all SMD M3216/1206 size 1% unless noted)

- 1 10MΩ (code 106 or 1005) 2 10kΩ (code 103 or 1002)
- 2 1MΩ (code 105 or 1004) 16 1kΩ (code 102 or 1001)
- 3 100kΩ (code 104 or 1003)

Substitutions for SMD LED version (replaces the parts marked with •)

- 1 double-sided PCB coded 08103242, 59.5 × 26mm
(instead of PCB coded 08103241)
- 7 blue 3mm high-brightness SMD M3216/1206/SMA LEDs (LED1-LED7)
- 7 red 3mm high-brightness SMD M3216/1206/SMA LEDs (LED11-LED17)

Kits available from the SILICON CHIP online shop (see p78 for more details):

❶ Mini LED Dice kit with through-hole LEDs (SC6849; ~£10.50 + P&P)

❷ Mini LED Dice kit with SMD LEDs (SC6961; ~£10.50 + P&P)

Kits include everything in the parts list except the cell. Prices include VAT but not postage.

cause of faults is a bridge between IC pins that's near the body of the IC, making it hard to spot.

Assuming it's working, check that both dice show valid numbers (refer to Fig.1). Press S1 and roll the dice, then check again that the states are valid. Repeat until you have seen all six numbers on both dice.

If any of the dice don't look right, check if it's because one or more LEDs are not lighting. If so, they might be connected backwards, have bad solder joins or (in rare cases) be duds. If all the LEDs are lighting but some of the patterns are wrong, check for solder bridges between IC pins or between components.

About 30 seconds after pressing S1, you should notice the LEDs fading out. Typically the blue ones will fade out and switch off before the red ones due to their higher forward voltages. After about 90 seconds, the LEDs should be off and the circuit is in a low-power state. Pressing S1 should switch it back on and roll the dice again.

Note that it's possible to get a short roll with a short press of S1. The results should still be random, but if you want to be sure, hold it down for a half a second or so rather than just pressing it.

Final assembly

If you are fitting the vibration sensor, remove the cell and bend its leads at right-angles to fit the PCB pads. We suggest doing this with two pairs of fine-nosed pliers to avoid damaging the sensor by applying too much force to the lead where it enters the sensor body. Lay it over the rectangle in the top-left corner of the board, solder it in place and trim the leads.

Now reinsert the cell and shake the board. It should switch on and roll the dice. They should roll every time you shake it.

Insert a short Nylon M2 machine screw through the small hole in the PCB, with the head next to the coin cell, and add a Nylon hex nut on the back. Do it up tightly, then trim off the

excess screw shaft length with side cutters. While it is almost impossible for children to remove the coin cell (due to the holder's tightness), it provides an extra layer of safety against especially keen toddlers.

Finally, slip a length of ~35mm diameter clear heatshrink tubing over the whole assembly, shrink it down (try to spread the heat out, rather than heating just one area) and trim the ends. That will protect it from moisture, dust, shorting against anything metal etc. To change the cell, cut it off and shrink on a new piece.

You can use the board without the heatshrink tubing but be aware that, as parts of the circuit operate at fairly high impedances to improve the battery life, your skin resistance (which can be well under 100kΩ) can mess with its operation. So it's better to sleeve it.

I noticed when I encapsulated the prototype, because the board got quite hot, it activated and the dice started rolling really fast. It went back to normal when it cooled down. I put this down to increased leakage through the Mosfet due to heat, providing enough current for the circuit to run, along with changes to the schmitt-trigger thresholds affecting the oscillator speed.

Also, if you are using the vibration sensor, its operation could be affected if it is squeezed too tight. I noticed a slight reduction in sensitivity but that could probably be fixed by adding a small slit in the tubing near the sensor to relieve the pressure on it. Alternatively, try to avoid shrinking the tubing fully in that area.

Conclusion

We aren't sure whether the randomness of our Dual SMD LED Dice is sufficiently good to run a tournament, but it should be fine for casual game playing and it's a conversation piece compared to regular dice. It also demonstrates what you can achieve with some very simple digital logic!

If using the vibration sensor, it probably isn't a good idea to keep it in a bag, a pocket or a vehicle as it might use up its battery quite quickly. **PE**

Coin Cell Precautions

Even though we have added protections such as the locking screw, it is best to make sure that children do not use this device unattended.

Ge semiconductors

Small-signal PNP transistors
NKT214F, OC57, OC59,
2N1377, 2N525 **£1.00**

Low-noise PNP transistors
GET106 **£2.50**

Small power PNP transistors
AC153, AC153K, AC188 **£1.50**

Small power NPN transistors
AC176, AC141K **£2.00**

High-voltage PNP transistors
OC77, CV7001 **£1.50**

TO3 PNP power transistors
OC22, CV7054 (OC23), OC25,
OC35, OC36, AD143, AD149,
AD161, AD162 **£2.00**
AD140 **£3.50**
AD149, AD161/2 matched pair **£5.00**

RF PNP transistors
OC41, OC42, 2SA12, 2SA53, AF124,
AF178, GET872A **£1.50**

NPN transistors
OC139, OC140, ASY73 **£2.00**

Diodes
CV7049 (OA10), CG92 (OA91) **£0.50**

Si semiconductors

Diodes
ZC5800 RF Schottky **£0.20**

Low-noise silicon transistors
BFW16A, 2SC3071, 2SC3068,
2SA1016K, 2SC2362K,
2SA970BL **£1.50**
2SC2204, 2SD655, BC550C **£0.50**
ZTX651 **£0.30**
ZTX751 **£0.50**

RF transistors (suitable for Theremin)
BF199 **£0.50**

Audio power MOSFETs
Exicon 10N20, 10P20 **£6.50**
Hitachi 2SJ99, 2SK343 **£3.50**
Hitachi 2SJ56, 2SK176 **£8.50**

JFETs
BFW11, BFW10, TIS73L, J177, J113,
U1994, U1898, 2SJ176, J201 **£1.00**
J175, J176, J112, J111, 2N3820,
2N5467, BF244, 2N5460, J230 **£0.60**

Small power output/driver transistors
2SB649A, D669A, 2SA1208,
2SC2910, MJE253G, MJE243G,
2SA1725, 2SC4511 **£1.20**
BD139, BD140, BD135, BD156, BD435,
BD436, MPSA63, BCV46 **£0.50**

MOSFETs
ZVP2106A **£0.30**
Dual-gate 3SK45, BFS28 **£1.50**

Metal-cased transistors
BC143 **£0.35**
2N1711 **£0.50**
BCY71 **£0.30**
BC109C **£0.60**

Amplifiers
LM318 high-speed op amp **£0.35**
µA709 metal-cased op amp **£2.00**
5534H metal-cased op amp **£3.00**
LM384, TDA2030A, **£3.00**
TDA2050V **£1.50**

TAA435 (Mullard power amp
driver for AD161/AD162) **£1.20**

Synthesiser ICs

That/dbx 2180 VCA/VCF **£6.00**
CA3080 VCA/VCF (vocoder) **£2.80**
CA3280 VCA/VCF **£4.00**
LM13600/ LM13700 VCA/VCF **£2.00**
CA3086 transistor array **£1.00**
PT2399 echo/delay **£1.50**

Dual transistors

2N2639, 2N2223, 2N2910 (NPN) **£4.00**
E401 (JFET Moog) **£4.00**
2SK2145-Y dual JFET **£0.80**
2N5564 JFET **£8.00**
HN3C51F, HN3A51F **£1.00**
DMMT3904/6, HN1A01F, **£1.00**
HN1C01F **£0.50**

Loudspeakers

PE Mini-Monitor
Volt PE165 6.5-inch woofer (each) **£85**
Morel MDT29 tweeter (each) **£25**
Kit pair of PE165/MDT29 plus
Volt crossover parts and PCBs **£299**
Monacor DT-28N tweeter (each) **£35**
Vifa 19mm BC20SC15-04
tweeter (each) **£15**

Volt crossover inductors
1.2mH, 1.5mH, 2mH, 2.7mH, 0.5mH
(tapped at 0.3mH) (each) **£5.85**

*Fully assembled and tested high-quality
speaker prototypes – ask for details*
LS3/5As and other similar speaker
systems (pair) **£200-£350**

Fully tested reclaimed speakers
Vifa BC14 5-inch woofer (each) **£10**
Vifa TC26 1-inch tweeter (each) **£10**

Low-price speakers
Philips 4-inch 4070 **£2.00**
EMI 10x6-inch, 30Ω Alnico **£7**
64mm 64Ω neodymium **£1.20**

Capacitors

Note '10/63' denotes '10µF 63V'.

Polyester
3.3/100, 4.7/250, 4.7/63 **£1.00**
5.6/63, 8.2/63, 10/63 **£2.00**
Mullard 'Mustard' C296 0.22/400 **£2.00**

Polycarbonate
Axial 2.2/63 1%, 4.7/160, 6.8/63 **£1.00**

Radial
6.8/160V, 10/63 **£2.00**
Reclaimed 22/63 **£2.00**

Polystyrene
Philips 1% 4.7nF/160, 6.2nF/500,
12nF/63, 22nF/63, 110nF/63,
24nF, 2nF **£1.00**
RIFA 1% 100nF/100, Suflex **£2.00**
90.9nF 0.5% **£2.00**
Suflex 2.5% 10nF/63 (rad. or ax.) **£0.50**

Radiation resistant
Siemens cellulose acetate MKL
2.2/25 **£0.80**

Electrolytic – Mullard blue 017 series
10/25, 22/25, 100/10 **£0.50**
150/40, 470/40, 1000/40 **£1.00**

Tantalum – axial metal cased
22/50, 47/35, 68/25, 100/20, 120/10,
150/16, 220/10, 330/6 **£2.00**
22/35, 33/35, 47/20, 68/15, 100/10,
150/6 **£1.25**
4.7/50, 6.8/35, 10/25, 10/35, 22/15 **£1.00**

Axial moulded-case tantalum
Kemet axial 6.8/10 **£0.30**
Kemet radial 33/10 **£0.40**
STC radial 100/20 **£1.50**

Tantalum bead
22/50 **£1.00**
470/3 **£2.00**
680/6.3 **£3.00**

Wet tantalum
220/25 axial **£3.20**
Castanet button 140/30, 470/3 **£3.20**
Hughes 540/10 **£3.20**

Bipolar
Hermetic bipolar tantalum 16/35 **£3.20**
Elcap axial 10/50 **£0.50**
Generic radial 100/16, 470/35,
100/ 63, 22/35, 4.7/35, 220/16 **£0.50**

Philips solid-aluminium (axial) 121/123
47/16 **£1.00**
330/6.3 **£2.00**
100/35 **£4.00**

Philips Pearl 122 series (radial)
10/16 **£0.30**

Silvered mica (radial)
1nF/500 1% **£1.00**

Trimmer capacitors
Vishay plastic-film 4-40pF **£0.80**
Vishay plastic-film 5.5-45pF **£1.00**
Vishay plastic-film 5-80pF **£1.20**
Vishay plastic-film 10-250pF **£2.00**
Mica 1-12pF, 2-40pF **£1.00**

Audio transformers and inductors

Eagle transformers
LT44, LT722 driver, LT700, LT723
500Ω output **£2.50**
LT30 500mW output **£3.50**
5:1 interstage **£1.50**

Repanco
T/T3 splitter transformer **£4.00**
CH2 5mH RFC **£2.00**

Balanced output transformer
Vigortronix 600Ω VTX-101-007 **£10**
Vigortronix 600Ω VTX-101-3001 **£10**
Vigortronix 600Ω VTX-101-3002 **£15**
Gardners 150Ω **£10**
Reclaimed BBC LL74/MPC
nickel core 600Ω **£12**
Low-level 1:1 600Ω £4

Reclaimed mic input transformer **£15**

Inductors
82µH, 4.7mH, 100µH, 270µH,
10µH, 14µH (low Z) **£0.50**
7-inch ferrite rod with MW
and LW windings **£3.00**

Special resistors

Bourns wire-wound trimmer
5kΩ 3059 JM panel-mount **£2.00**
Thermistor RA53, R13 **£4.00**
A13 **£2.00**
Thermistor CZ1, CZ6 **£1.50**

Holco H2 2.2MΩ 1W, 1% **£1.00**
Welwyn 1GΩ 2W **£1.00**
5k Bourns 3321H cermet trimmer **£0.50**

Potentiometers

Bourns 81
25kΩ lin cermet **£2.00**
25kΩ lin, 5kΩ lin conductive
plastic **£3.00**

Bourns 91
10k dual-gang lin or log **£10.00**

Plessey moulded-track
5kΩ log with switch **£3.00**
50kΩ A/log **£3.00**
Mil 250kΩ lin dual **£5.00**

Alpha
16mm 4.7kΩ A/log **£0.80**
220kΩ A/log **£0.80**
10kΩ lin centre-detent dual-gang **£1.50**

Allen Bradley J series/Honeywell
10kΩ lin 1 million cycles **£5.00**
Blore Edwards AB 45 dual 5kΩ
A/log with switch **£3.50**
Alps RK9 dual-gang 5kΩ RD law **£4.00**
Alps 50k log tapped motorised
stereo **£5.00**
BI P260 500kΩ log conductive
plastic 1 million cycles **£2.00**
Colvern wire-wound 100kΩ or
50kΩ dual-gang 3W **£5.00**
Mil Spec hermetic 10Ω **£8.00**

Miscellaneous

PCBs
Pocket Theremin (*EPE*, 1996) **£2.00**
Elysian Theremin (*EPE*, 1996) **£6**
Synth VCF, VCO (*EPE*, 2017) **£3.00**
48V PSU (*EPE*, 2019) **£3.00**
SMT dual transistor adapter **£0.40**

Relays
STC hermetic 4190 24V DPDT **£4.00**

Switches Blore Edwards/Plessey
MH1 4-pole 3-way **£3.00**
Elma 01 2-pole 6-way **£6.00**
3PDT guitar pedal stomp switch **£2.50**

Valves
EC88, PCL84 **£3.00**

Contact

Jake Rothman
Manchester House
Marker Street, Craven Arms
Shropshire SY7 9NN
(visit by appointment)
+44 (0)1597 829102
jrothman1962@gmail.com

Minimum order £5.00 inc post
Quantity discounts negotiable

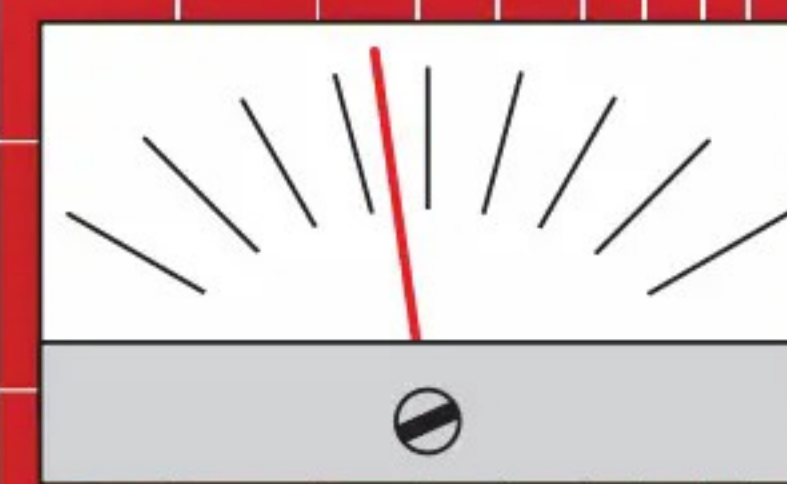
Payment

PayPal, bank transfer, cheques
(payable to "J Rothman", UK
pounds only)
No VAT payable

Postage

Small Jiffy bag £2.99
Small package £4.99
Big boxes and overseas at cost
– ask for a quote

AUDIO OUT



By Jake Rothman

A phantom-powered DI box using a JFET/BJT cascode

Following on from last month's column with the theme of unusual experimental audio circuits, I've dug another one out from my notebooks.

This direct injection (DI) box circuit is unusual in using a JFET and BJT in cascode to handle the relatively high phantom power voltage.

Electric guitars need to be loaded with a high impedance, over $1\text{M}\Omega$. Otherwise, they sound dull and the high-resistance volume and tone controls don't work properly. This is at least partly because they evolved during the valve era, and guitarists are conservative musicians.

I've tried to persuade some musicians that a low-impedance balanced output is the way to go for the best sound quality, but they are wedded to their legacy hardware. Also, the idea of drilling a 22mm hole in their beloved instrument for an XLR socket is anathema to them.

The solution to this dilemma is a direct inject (DI) box, which provides a high-impedance input for the guitar and a balanced, low-impedance output to feed a low-to-medium-impedance input, like a mixing desk input. Mixing desk inputs typically have an input impedance of $5\text{--}10\text{k}\Omega$.

Recording guitar

An important factor in getting a good electric guitar sound in the studio is to mic up a valve guitar amp and mix it with the DI input to the desk, as illustrated in Fig.1. Guitars do not sound good recorded directly. They need the overdrive distortion of a valve amplifier and the multiple breakup modes of a light-weight paper speaker cone.

Multiple effects are also used on both channels to give a massive sound with stereo spread. This combination of two mains Earthed inputs would normally cause a hum loop between the amp and

the mixing desk, so all useful DI boxes have a transformer-isolated balanced output. A direct feed from the guitar to the amp has to be provided, as shown in Fig.2.

Going active

Attempts were made in the 1980s to make active, transformerless DI boxes but they never quite eliminated the hum because the output has to be referenced to the ground of the guitar amp. Most DI boxes are passive devices, using a 10:1 step-down transformer (Fig.3), which needs expensive Mu-metal shielding and gives a poor frequency response.

By using an amplifier to drive a lower-ratio transformer, as done here, these problems are eliminated. The downside is that a power supply is needed. This is normally a 9V block (PP3) battery, which is expensive and has a short life. The solution used here is to take advantage of the 48V phantom power available on most mixing desks and computer studio interfaces.

Amplifier design

Since phantom power is a relatively high-voltage, low-current supply, a discrete amplifier design is preferable to an op amp, as well as being more fun to design. A JFET input is a good idea for a high input impedance, since it gives lower noise with the high source impedance of the guitar. Most low-cost JFETs are limited to 30V, which gives a good excuse to employ a cascode.

A cascode is where two transistors

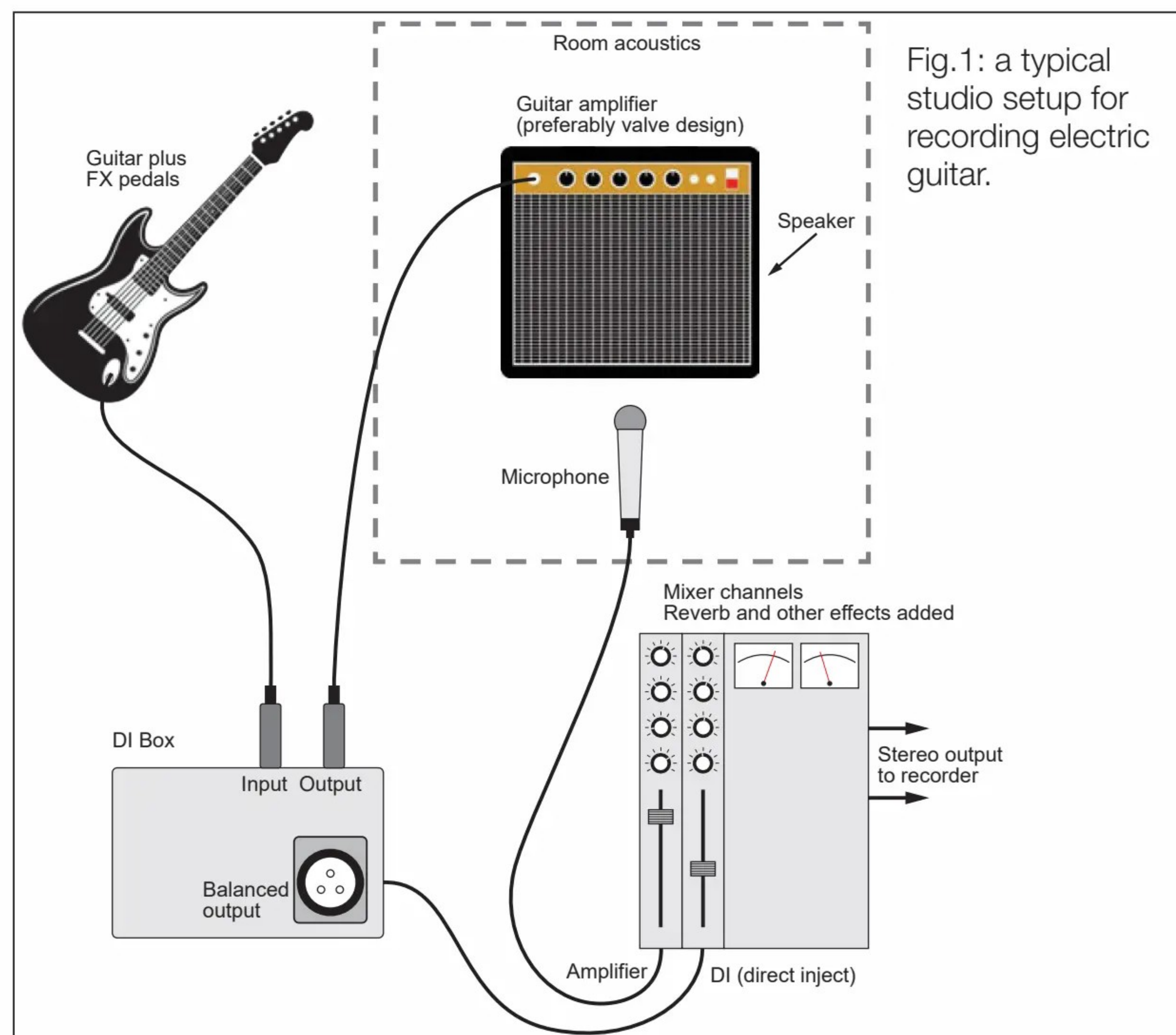


Fig.1: a typical studio setup for recording electric guitar.

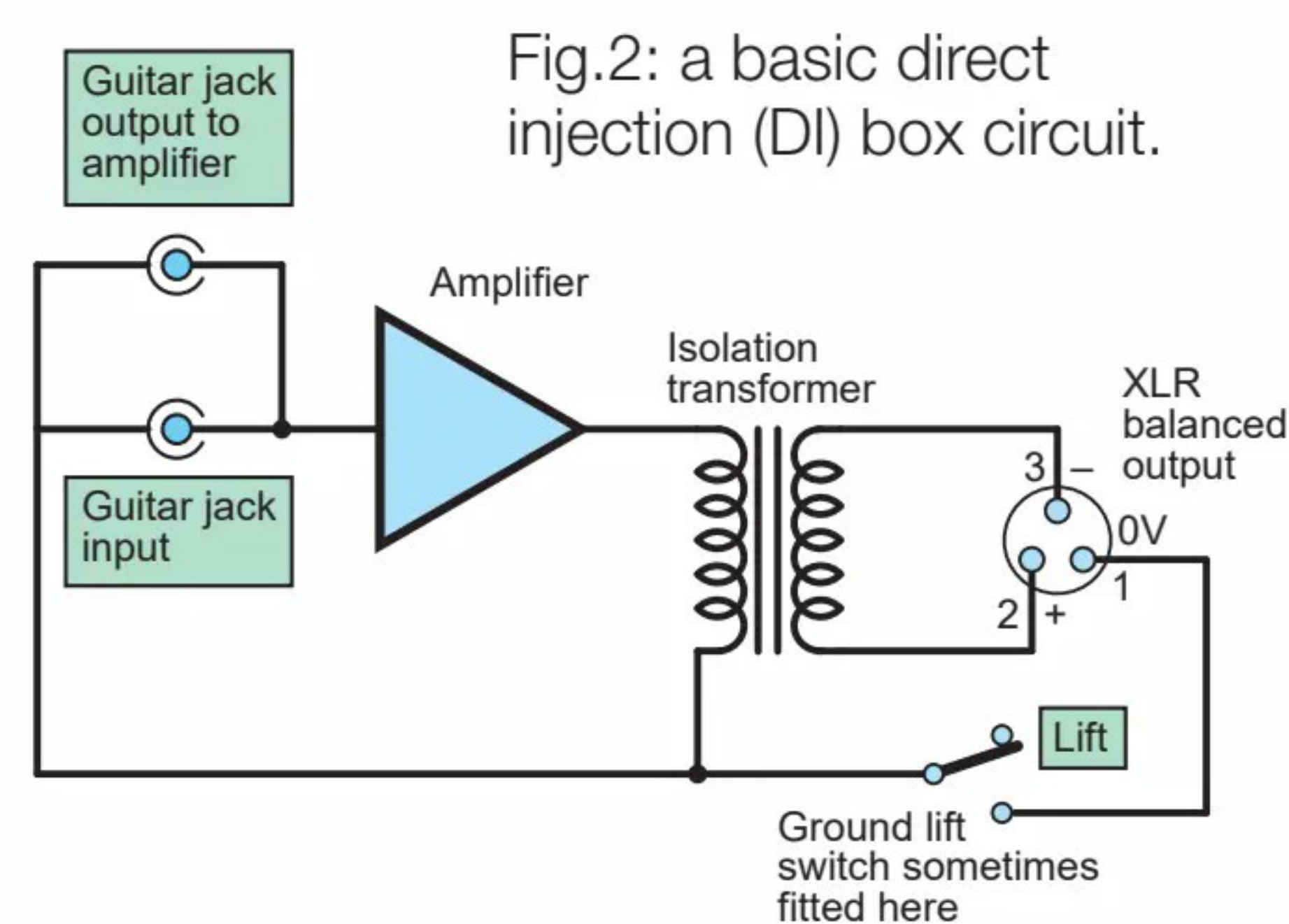


Fig.2: a basic direct injection (DI) box circuit.

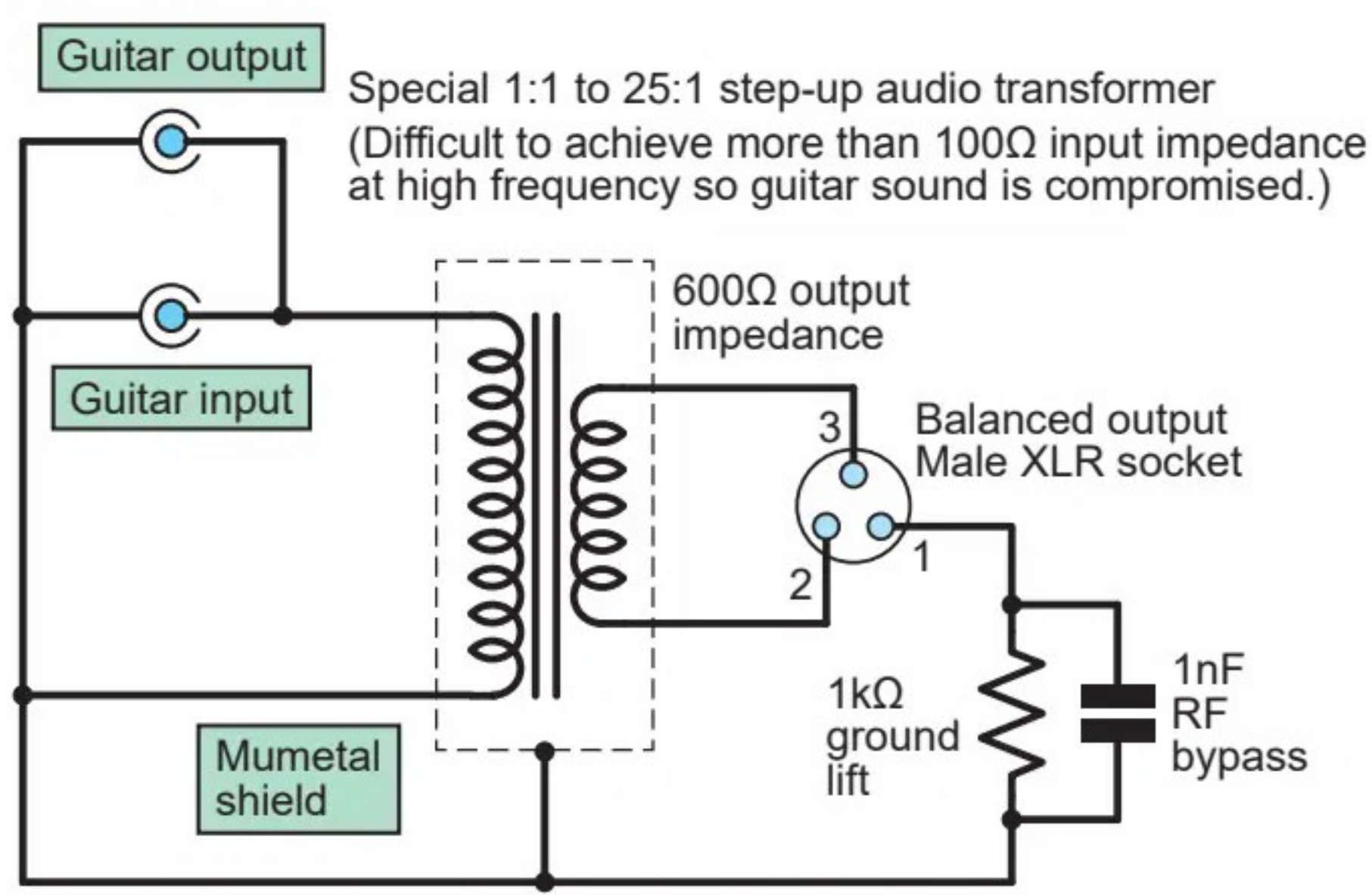


Fig.3: a passive DI box circuit.

are placed one on top of the other, in series, as shown in Fig.4. The supply voltage drop is shared by both devices, allowing a low-voltage device, in this case the JFET, to be used for the lower input device.

There's nothing stopping you from cascoding bipolar junction transistors (BJTs), or even a combination of BJTs, JFETs, Mosfets or even valves. For guitar practice amps, I like to use a JFET at the bottom with a valve on top. A dual-gate Mosfet can be considered a 'cascode in a can'!

There are other advantages to the cascode configuration that are not of great importance here, but are worth knowing. This simple circuit gives an opportunity for experimentation.

The Early effect

The cascode was popular in top-notch Hi-Fi amplifiers from the 1970s. The PE Gemini designed by Ferranti engineers was the first in a magazine, and it still sounds and measures very well today. A cascode was necessary to avoid an odd distortion arising in the main voltage amplification stage (VAS) transistor when the emitter-collector voltage (V_{CE}) swing was almost rail-to-rail.

This large swing caused the transistor to modulate its own gain and junction capacitance by a mechanism called the Early effect (named after James M. Early), where the thickness of the active region of the semiconductor varies with applied voltage. By using a cascode, the voltage across the lower transistor is effectively fixed, at a few volts, preventing this modulation.

The effective capacitance is also reduced, improving high-frequency performance.

This configuration is still popular in RF amplifiers and deflection amplifiers in cathode ray tube (CRT) oscilloscopes. The Early effect is less of a problem with modern VAS transistors, many of which were originally developed for deflection transistors in CRT monitors in the 1980s. The data sheets of such transistors will often display a very flat h_{FE} vs V_{CE} curve.

Those made by Toshiba, Sanyo and

Fig.4: four examples of the basic cascode configuration. Both active devices can be of any type (bipolar, JFET, MOSFET or valve).

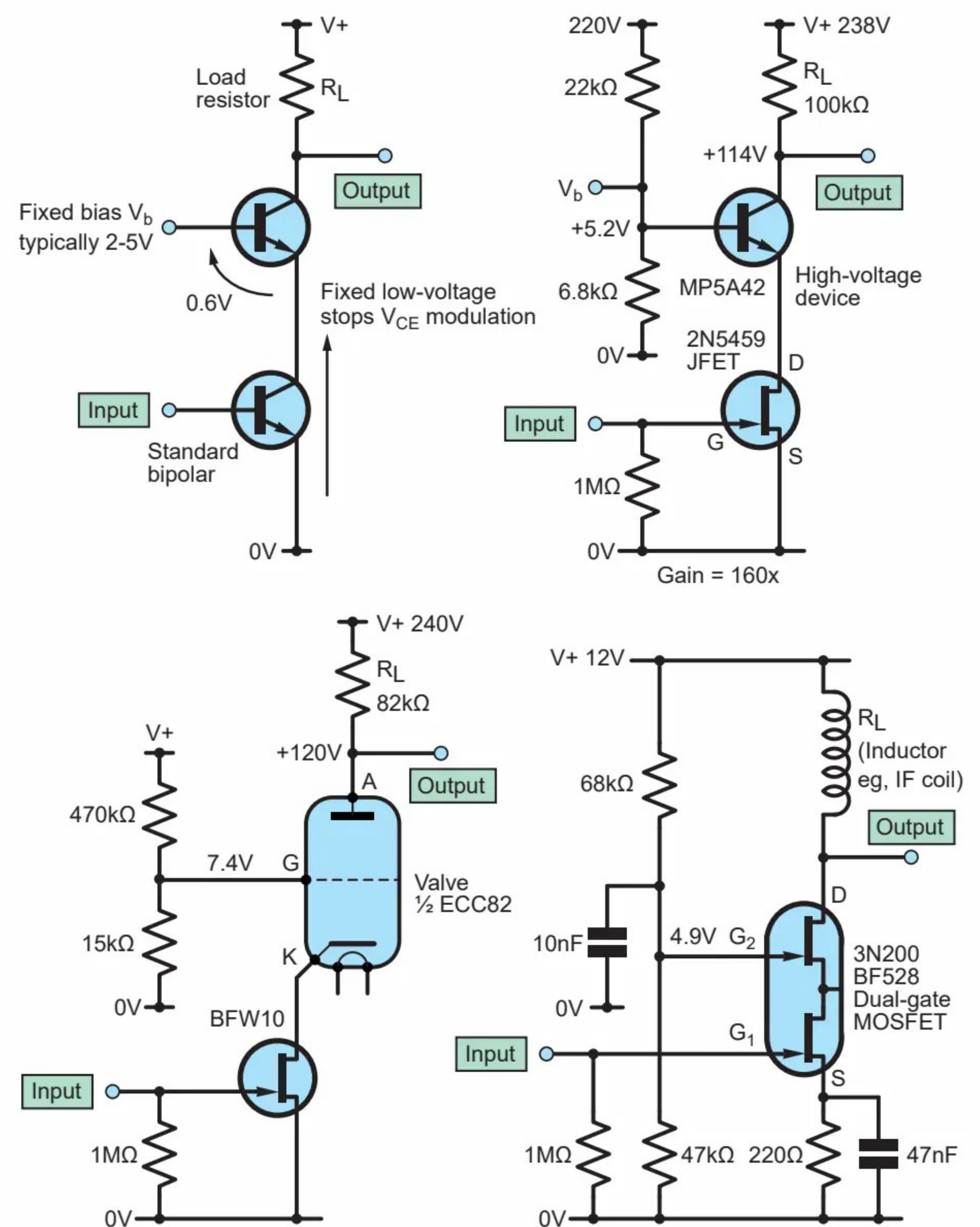
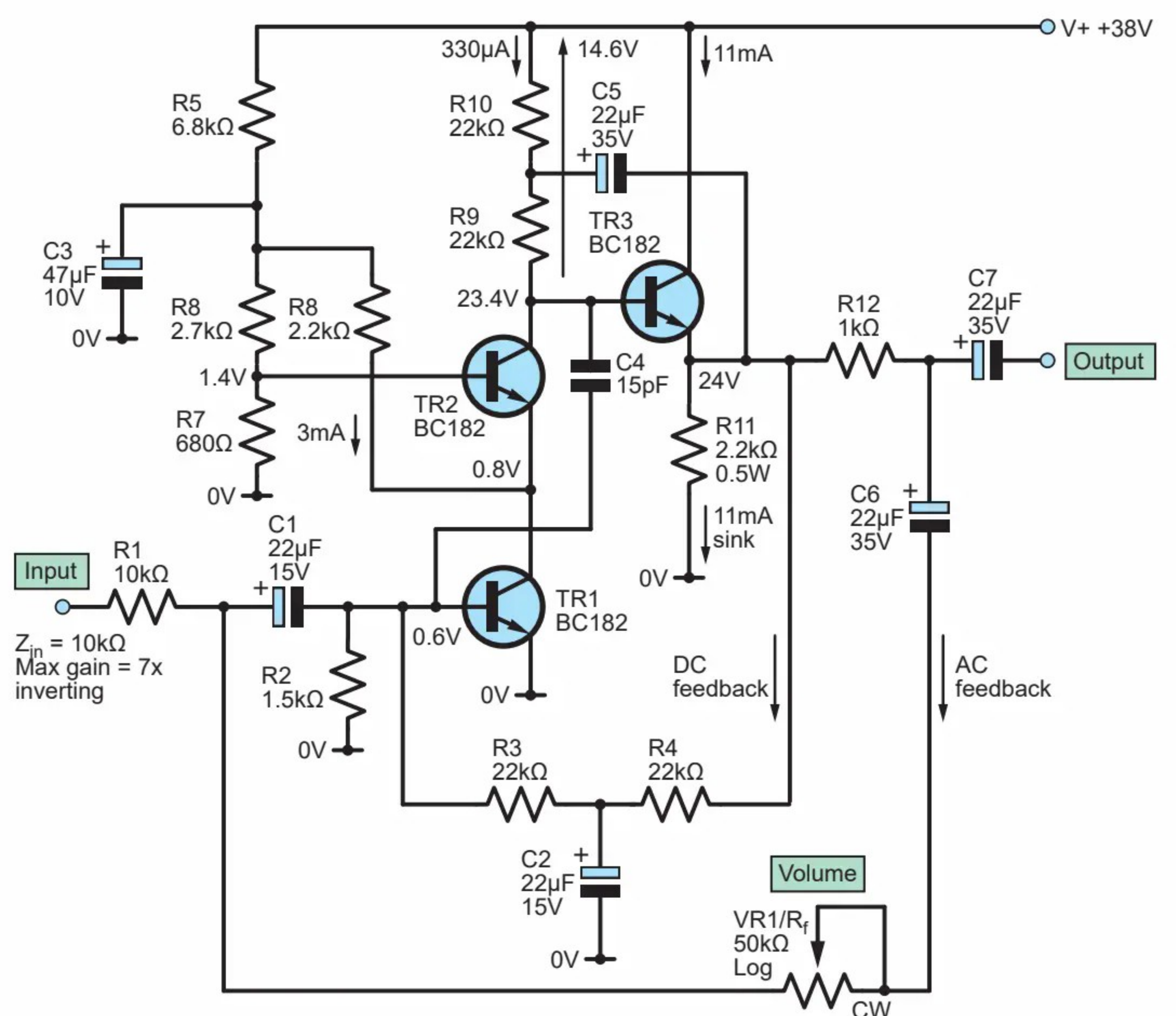


Fig.5: Douglas Self's cascode amplifier circuit from *Wireless World*, February 1979.



Hitachi are considered the best. In the quest for maximum power, the cascode is used less in discrete power amplifiers because it loses a couple of volts of headroom for a given supply voltage. It is still used frequently in high-quality JFET audio op amps, where there is virtually no limit on the number of transistors that can be used.

The first time I saw a cascode was in Douglas Self's 1979 *Wireless World* pre-amp, where it was employed in the main active gain stage (Fig.5). This three-transistor circuit gave the

same performance as the NE5534 op amp at a tenth the price (at the time). I upped the power rail on mine to the standard telecom/phantom 48V and used a Bourns 91 conductive plastic volume control.

An interesting tweak is the extra resistor to boost the current to the lower transistor (R8), since this stage usually saturates first. Sadly, this lovely pre-amp was stolen one jazz night at the Hebden Bridge Trades Club in 1993. I've been meaning to build another ever since.

“Root canal” fuzz

One overlooked problem with the cascode is its horrid overload waveform, modern power amps are expected to clip cleanly. I use it here to give an unusual fuzz effect, where the top of the square wave starts collapsing in on itself (Fig.6), generating a rash of extra 'harsh' high-frequency harmonics.

A practical circuit

Finally, we get to a buildable circuit, shown in Fig.7. The input is AC-coupled by capacitor C1, while the combination of resistor R1 and capacitor C2 act as a low-pass filter to provide a degree of RF rejection. The input impedance is defined by resistor R2; I find that 2.2M Ω gives the best results with most guitars.

The input signal is applied to the gate of N-channel JFET TR1. This can be any low-current, low- I_{DSS} type, such as the 2N5457 shown here. The J202, J305 or BF245 are also suitable. The popular 2N3819 and similar types need too much current, which would overload the phantom supply.

Since the gate source voltage (V_{GS}) for a given channel current (I_D) varies so much between individual JFETs, preset VR2 is needed to trim for symmetrical clipping. Trimmers are bad for mass production, but interesting for tweaking. Some guitarists like asymmetrical clipping for the even harmonics it produces, similar to how a 12-string guitar has extra strings tuned an octave higher.

The upper transistor of the cascode is a high h_{FE} device because the partition noise is less. Since the voltage is shared, a low-voltage 30V transistor can be used, such as a BC549C. The cascode stage gives a gain of around 15 $\times$. The signal is then directly coupled to a second PNP BJT transistor, boosting the open-loop gain to around 400 $\times$. The final gain is controlled by negative feedback (NFB) through resistor R6. The minimum gain is around 2 $\times$ with the gain control pot, VR2, fully anticlockwise.

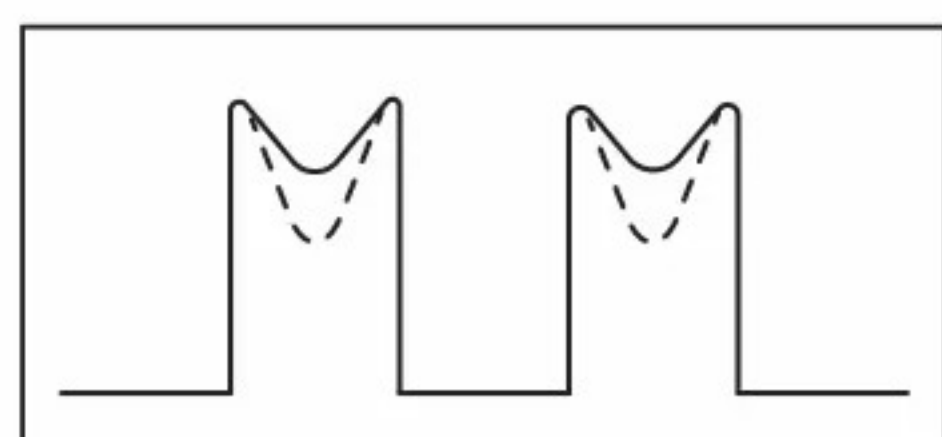


Fig.6: the strange clipping effect of this cascode circuit.

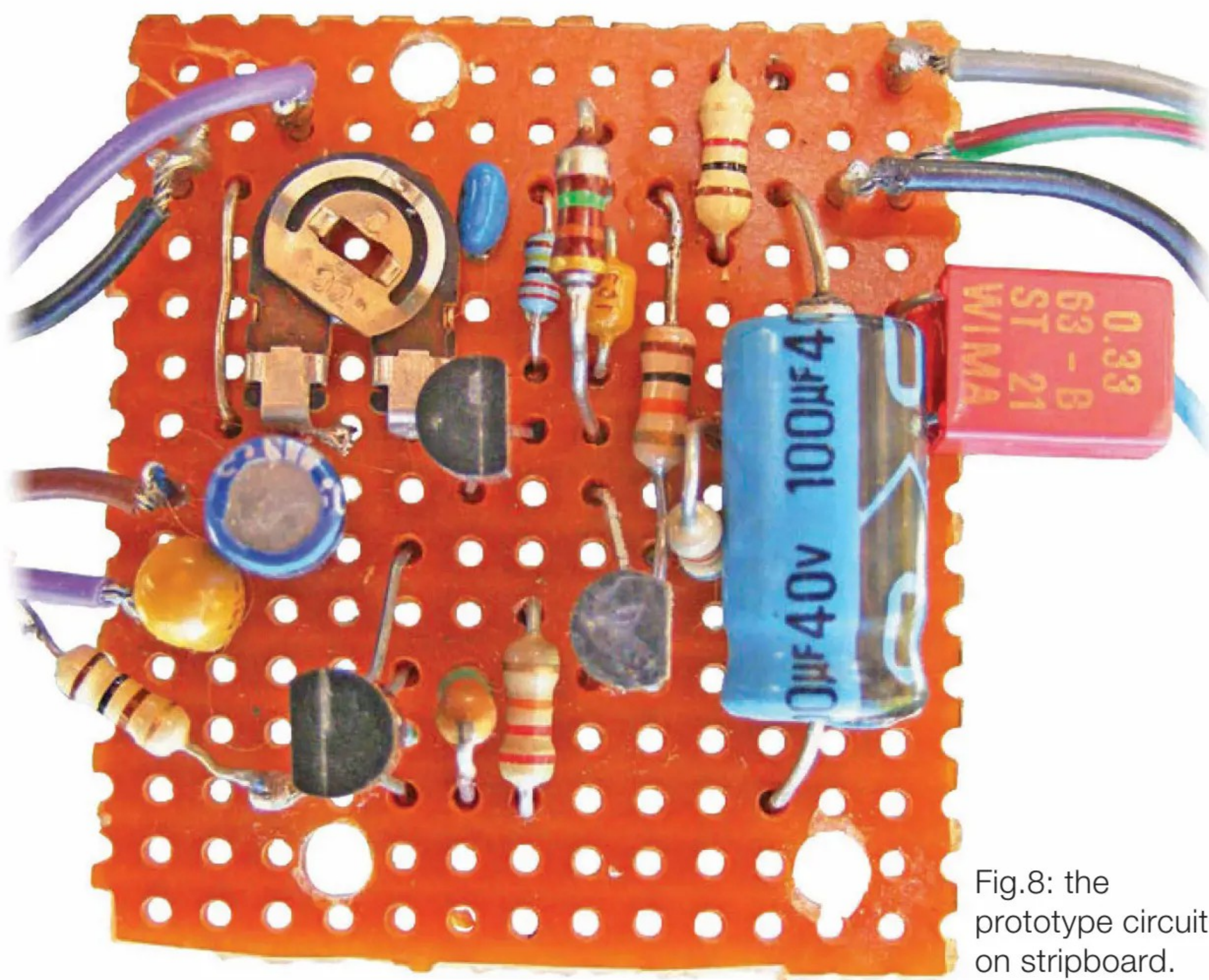


Fig.8: the prototype circuit on stripboard.

Note that the Vigortronix output transformer has a 2:1 step-down ratio, reducing the minimum gain to unity, while doubling the output current available.

The fuzz switch, SW2, just removes all NFB, giving full gain. A treble boost function is obtained by bypassing some of the NFB to ground via capacitor C3. This function is switched in with SW1. Interestingly, jazz guitarists like the effect, while rock guitarists hate it.

The whole circuit is non-inverting and the output transformer should be wired to ensure the phasing is correct. Most mixers have a phase invert button.

If yours doesn't, it may be worth incorporating one on the output transformer, since some guitar amps are inverting.

Power supply

For the power supply, a centre tap on

the output transformer is needed. The phantom power is fed through 6.8k Ω resistors mounted in the mixer (shown faded on the diagram). This supply is decoupled by capacitor C7. Pin 1 (0V) on the XLR connector is fed through ground-lift resistor R11 and bypassed for RF by capacitor C8, then connected to signal ground.

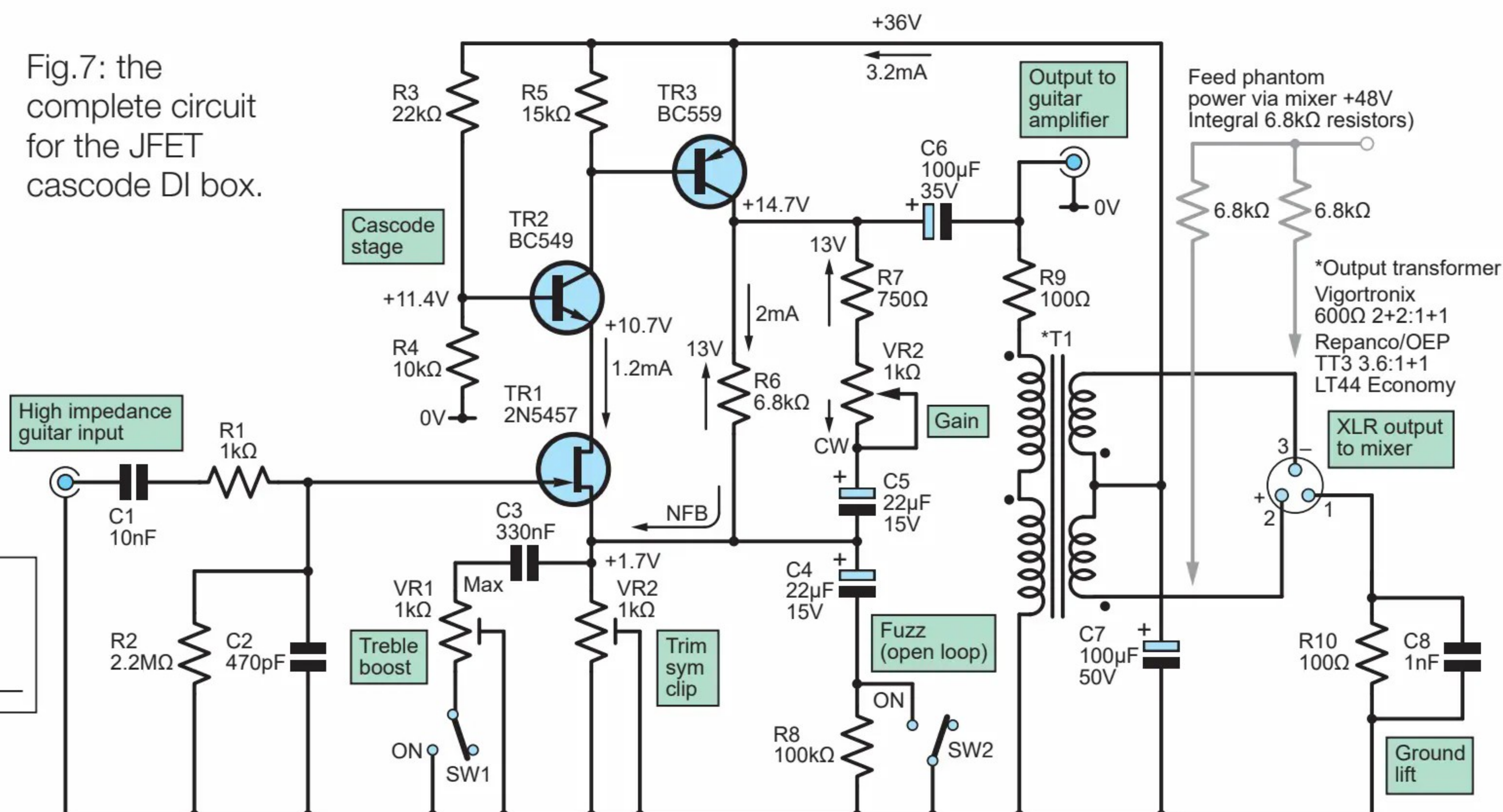
The internal connection between the phantom power supply 0V and op amp power supply 0V in the mixer could cause an Earth loop. It might be worth fitting a ground-lift resistor here as well.

Originally, phantom power was designed for microphones, which have no mains Earth connection. This is an area that needs further investigation.

Construction

There is not much to say here since the circuit is experimental, but it lends itself to stripboard construction, as

Fig.7: the complete circuit for the JFET cascode DI box.



NEW!

5-year
collections

2019-2023

All 60 issues from Jan 2019
to Dec 2023 for just **£49.95**

PDF files ready for
immediate download

2018-2022

All 60 issues from Jan 2018
to Dec 2022 for just **£49.95**

PDF files ready for
immediate download

2017-2021

All 60 issues from Jan 2017
to Dec 2021 for just **£49.95**

PDF files ready for
immediate download

2016-2020

All 60 issues from Jan 2016
to Dec 2020 for just **£44.95**

PDF files ready for
immediate download

See page 41 for further
details and other great
back-issue offers.

Purchase and download at:
www.electronpublishing.com

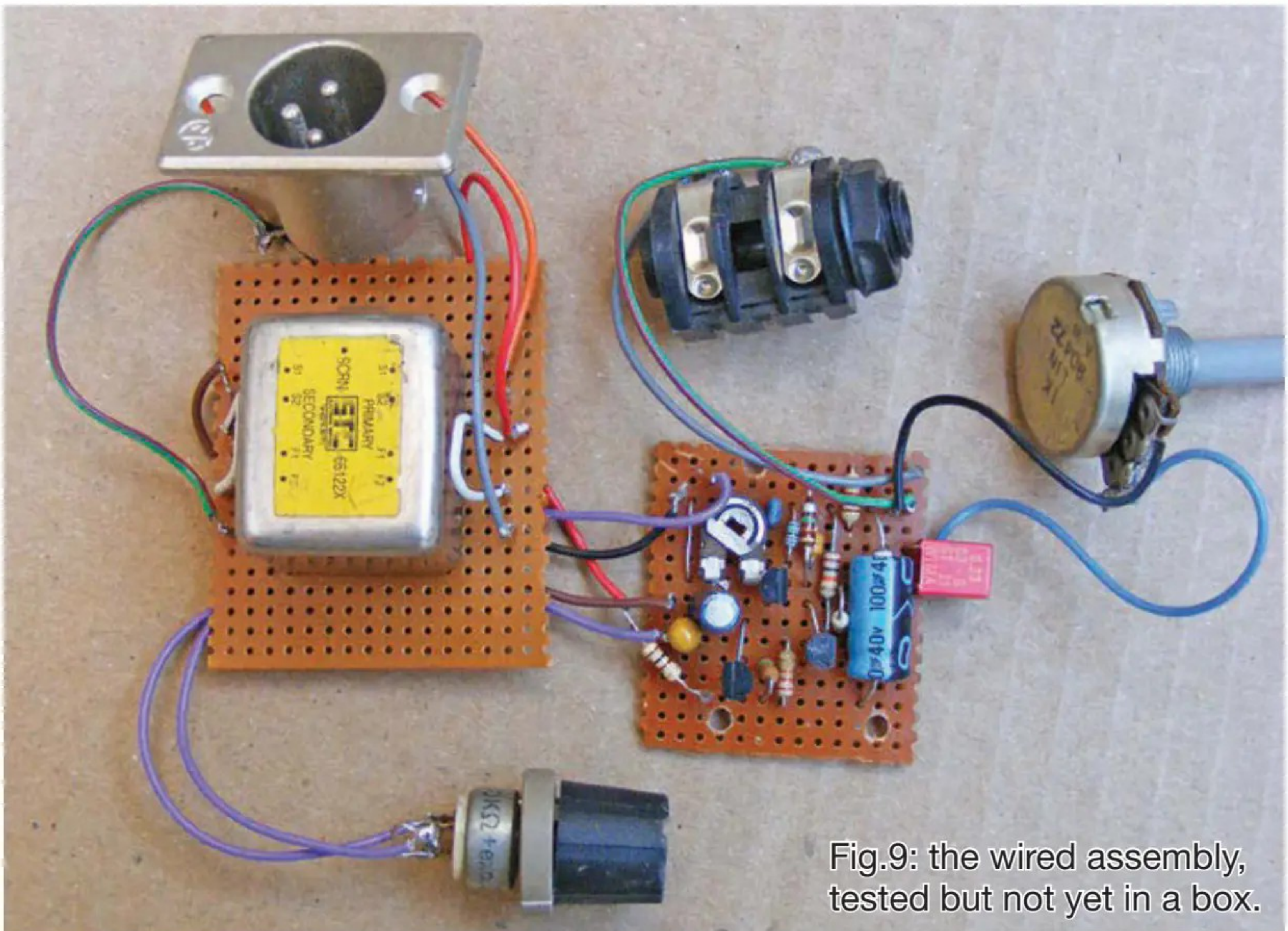


Fig.9: the wired assembly,
tested but not yet in a box.

shown in Fig.8. The output transformer is quite heavy, so it is a good idea to use a separate mounting PCB for it, such as the ones described in *Audio Out*, November 2022.

Fig.9 shows the PCB wired up to the transformer, connectors and pots before they were installed in a case. A 6.35mm TRS jack could be substituted for the XLR connector.

I was unable to create a proper PCB

or detailed construction plans since I just moved and haven't completed my horrendous house/workshop repairs (the old wiring needs a lot of attention!).

While on the subject, AOShop is now located at Manchester House, Market Street, Craven Arms, Shropshire SY7 9NN. The phone number and email are still the same as before: 01597 829 102 & jrothman1962@gmail.com. **PE**

Parts List – Phantom-powered DI box

- 1 2x2-inch (~50x50mm) piece of Veroboard
- 1 Vigortronix VTX-101-002 2.4kΩ to 600Ω transformer, 2+2:1+1 (T1)*
- 1 male XLR 3-pin chassis-mount connector
- 2 6.3mm mono jack sockets
- 2 miniature SPST toggle switches
- 1 10kΩ logarithmic potentiometer (VR1)
- 1 1kΩ trimpot (VR2)

Semiconductors

- 1 2N5457 N-channel JFET, TO-92 (TR1)
- 1 BC549C NPN transistor, TO-92 (TR2)
- 1 BC559 PNP transistor, TO-92 (TR3)

Capacitors

- 1 100μF 50V electrolytic (C7)
- 1 100μF 35V electrolytic (C6)
- 2 22μF 16V electrolytic, aluminium or tantalum (C4, C5)
- 1 390nF (preferred) or 330nF polyester (C3)
- 1 10nF polyester (C1)
- 1 1nF ceramic (C8)
- 1 470pF NP0/C0G ceramic (C2)

Resistors (all ±1% ¼W)

- | | | |
|------------------------|-------------------------|-----------------|
| 1 2.2MΩ (R2) | 1 15kΩ metal film (R5) | 1 1kΩ (R1) |
| 1 100kΩ (R8) | 1 10kΩ metal film (R4) | 1 750Ω (R7) |
| 1 22kΩ metal film (R3) | 1 6.8kΩ metal film (R6) | 2 100Ω (R9–R10) |

Qty	Value	4-band code	5-band code
1	2.2MΩ		
1	100kΩ		
1	22kΩ		
1	15kΩ		
1	10kΩ		
1	6.8kΩ		
1	1.0kΩ		
1	750Ω		
2	100Ω		

* Alternatives:

- | | |
|-------------------------------|---------------------------------------|
| Repanco TT3/OEP E187B 3.6:1+1 | Eagle LT722 1.4:1 (cheap) |
| Jaycar MM-2532 1:1 (cheap) | 600Ω 1:1 modem transformer (cheapest) |

I've been seeing robots on TV and in movies since I was a kid, but I doubted I would get to see humanoid models in factories, homes and care facilities in my lifetime. Then, when I least expected it...

"I've seen things you people wouldn't believe", as Roy Batty (played by Rutger Hauer) famously said in his "Tears in Rain" soliloquy towards the end of the 1982 movie *Blade Runner* (<https://youtu.be/NoAzpa1x7jU>). This movie, now considered a masterpiece, featured groundbreaking visuals, complex themes and a thought-provoking exploration of humanity and identity.

I know how Roy felt. I also have seen things you people (well, certainly any younger readers) wouldn't believe. For example, I hail from the days before anyone we knew owned a washing machine or a clothes dryer.

When I was around nine years old, I spent most of the weekdays during the summer holidays hanging out at my aunt's house with my cousin Gillian, who was one year younger than me.

Monday was "washday"; that is, the day that all the ladies did their laundry. My aunt had a massive mechanical device called a 'mangle' or 'wringer', featuring two big timber rollers and a large handle. Turning the handle made the rollers rotate, pulling the clothes through and squeezing any water out.

It was Gillian's and my job to crank the handle on the mangle. We took turns. We mangled and mangled until our arms ached, and then we mangled some more.

Later, in the afternoons, we used to watch the *Jetsons* on TV. I used to wish we had a robot maid like Rosie who could do the mangling for us. Of course, I knew Rosie was only a cartoon character. She wasn't real like the robot (simply called "Robot") in *Lost in Space*, but he wouldn't have been of much use either. The best he could do was wave his arms around and say, "Danger, Will Robinson!"

Why humanoid?

Robots come in all shapes and sizes and can do all sorts of things. For example, I recently saw an industrial robot replicating the actions of a Japanese master swordsman (<https://youtu.be/O3XyDLbaUmU>). I also saw a video on LinkedIn of two-armed robot on a wheeled base happily unpacking a dryer and folding the clothes (<https://pemag.au/link/ac52>).

So, what's our fascination with humanoid robots? It actually makes a lot of sense. First, human environments—like homes, offices, and factories—are (unsurprisingly) designed for humans. This means that robots with humanoid shapes can better navigate things like stairs and doorways, and can use tools, operate machines, and manipulate objects meant for human hands.

There's also the fact that humanoid robots can interact with humans more naturally, using familiar gestures, expressions, and speech. This facilitates their use in diverse roles such as being companions, caregivers, receptionists, and tutors, where human-like interaction enhances the experience.

Developed nations are increasingly grappling with the multifaceted challenges of ageing populations, necessitating comprehensive strategies to address economic, healthcare, and social impacts.

A major application for humanoid robots would be aged care, either in the home or in dedicated facilities. They could also help disabled people by performing tasks, providing companionship and aiding in rehabilitation.

My own dear old mum will be 95 this year. She's finding it increasingly difficult to move around her apartment. My brother lives nearby, and he visits her several times a day, but it would be fantastic if she had a humanoid robot to help by doing things like making cups of tea and preparing simple meals.

Not in my lifetime!

Way back in the mists of time we used to call the 1980s, when I became an engineer, I began to appreciate the problems involved in building a self-powered (that is, not tethered to the wall by a power cable) human-sized humanoid robot.

There are multiple aspects to this, including the robot's physical body, the advanced sensors and control systems required for it to perceive and interact with the world, and the 'brain' that tells it what to do.

I read a lot of science fiction, and I certainly believed that humanoid robots would be possible one day, but I also assumed that day would be a long time in the future—quite possibly long after I'd shuffled off this mortal coil and headed out to the next plane of my existence. How things have changed!

Several robot companies seem to have popped up over the past couple of years. A good example of the sort of thing I'm talking about is provided by the folks at 1X. They have a robot called EVE Industrial (<https://1x.tech/eve>).

Although EVE has a head and two arms with hands that can pick objects up and manipulate them, it's not humanoid in the full sense because it's mounted on a single leg that's attached to a mobile platform. Still, EVE can crouch down, reach up, and is well suited to working in factories and industrial facilities.

However, the robot that took my breath away is the recently announced NEO Gamma (<https://1x.tech/neo>).



An old-fashioned clothes mangle for drying wet clothes.

Source: www.pinterest.com/pin/457467274621825135/

Techno Talk

Max the Magnificent

This is a fully bipedal humanoid robot intended for use as a household assistant. “Oh, my goodness gracious me”, is all I can say.

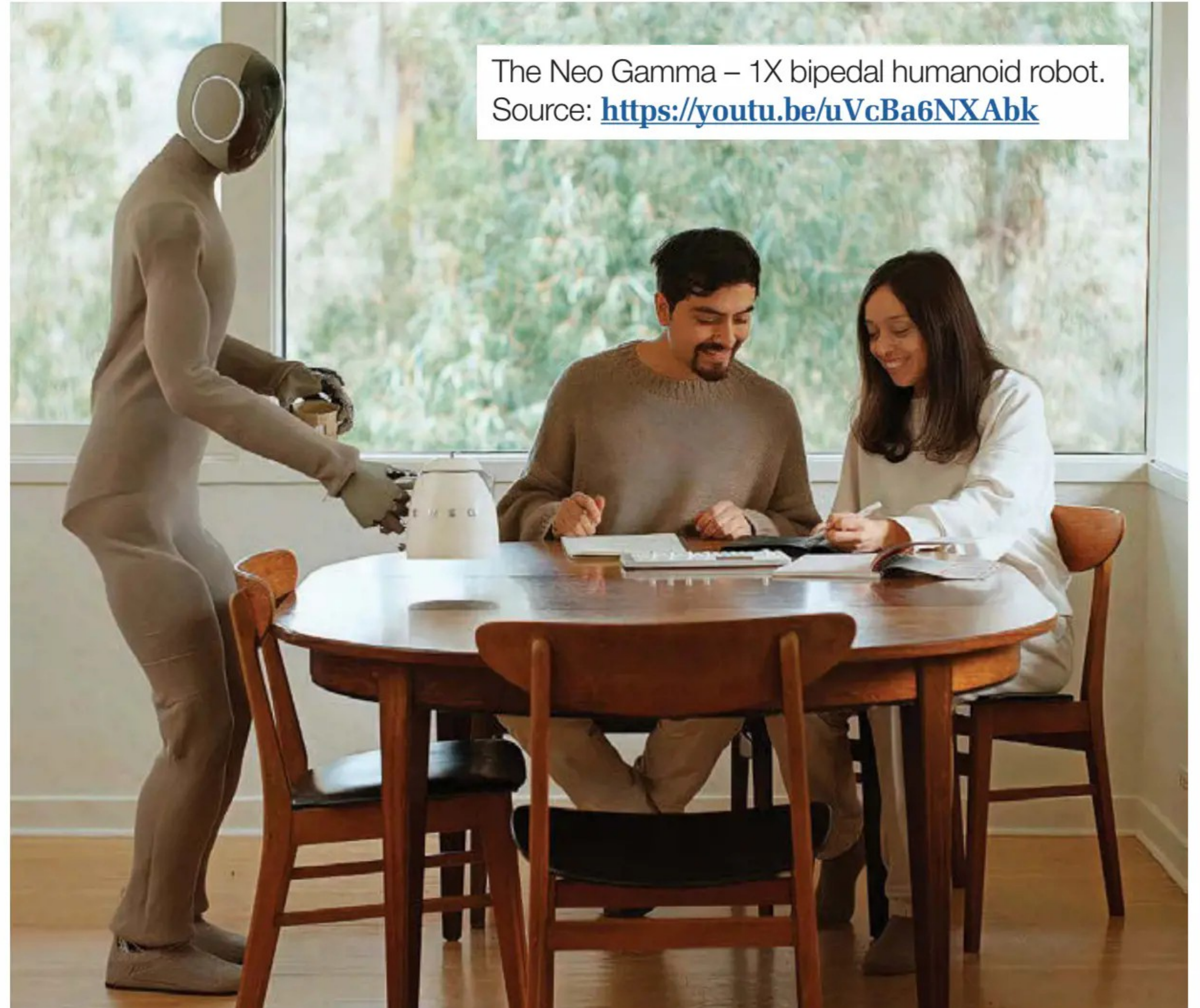
The Life of (a boy called) Clive

In my April column, I mentioned the fact that one of my hobby projects is writing a book called The Life of (a boy called) Clive that describes my life growing up as a kid in Sheffield in the 1960s. Well, I’ve finished!

This has taken over two years to complete, which is a lot longer than I expected. The problem is that I had so many adventures, like falling into a water barrel, falling out of a sixth-floor window, and falling off a cliff (I was a little scamp).

My next problem is to create accompanying images. I’m thinking of simple pencil sketches like those in Winnie the Pooh, perhaps with a splash of watercolour, like those in Into the Uncut Grass. The problem is that I can’t draw.

Happily, the folks at Kinara (kinara.ai) just sent me one of their Ara-2 USB modules that’s capable of supporting generative AI workloads such



The Neo Gamma – 1X bipedal humanoid robot.
Source: <https://youtu.be/uVcBa6NXAbk>

as the stable diffusion text-to-image generator.

It literally arrived as I was writing

this column and I cannot wait to try it out. I will, of course, report back later on how it goes. **PE**

GET THE
LATEST COPY
OF OUR
TEACH-IN SERIES
AVAILABLE
NOW!

ON SALE in WHSmith and other
independent newsagents

PRICE £8.99

(includes P&P to UK if ordered direct from us)

Teach-In 9 – Get Testing!

This series of articles provides a broad-based introduction to choosing and using a wide range of test gear, how to get the best out of each item and the pitfalls to avoid. It provides hints and tips on using, and – just as importantly – interpreting the results that you get. The series deals with familiar test gear as well as equipment designed for more specialised applications.

The articles have been designed to have the broadest possible appeal and are applicable to all branches of electronics. The series crosses the boundaries of analogue and digital electronics with applications that span the full range of electronics – from a single-stage transistor amplifier to the most sophisticated microcontroller system. There really is something for everyone!

Each part includes a simple but useful practical test gear project that will build into a handy gadget that will either extend the features, ranges and usability of an existing item of test equipment or that will serve as a stand-alone instrument. We’ve kept the cost of these projects as low as possible, and most of them can be built for less than £10.

FREE COVER-MOUNTED CD-ROM

On the free cover-mounted CD-ROM you will find the software for the *PIC n’ Mix* series of articles. Plus the full *Teach-In 2* book – *Using PIC Microcontrollers – A practical introduction* – in PDF format.



ORDER YOUR COPY TODAY: www.electronpublishing.com

WEIRD & WONDERFUL ARDUINO PROJECTS

Part 6: shift registers and 8-segment displays

Before we plunge headfirst into the fray with gusto and abandon, let's briefly set the scene. We are in the process of constructing retro gaming consoles that will make us the envy of our families and friends.

An early prototype is shown in Photo 1. Our main display is a 14×10 array of tricolour light-emitting diodes (LEDs). As of last month, we have this display up and running, although there remain optimisations we have yet to implement. Soon we will turn our attention to the 7-segment displays that can be used to present information like game levels and scores.

As we discussed in the February 2025 issue, we are going to use pre-built modules that contain 7-segment displays, 74HC595 shift registers and any ancillary components like current-limiting resistors. 74HC595s are the same shift registers we used to build the clock in our *Arduino Bootcamp* series (especially the December 2024 column).

By popular demand, the whole series is now available to purchase as a download, at <https://pemag.au/link/ac3d>

These modules are available from multiple suppliers, but the prices on AliExpress are hard to beat (<https://pemag.au/link/ac53>). They are available with two, three, or four digits. I've opted to use three two-digit modules on my console, but you can choose to use as many or as few of whichever types of these devices as you wish.

These displays can be daisy-chained together, so driving them all will require only three of our Arduino's pins. However, before we add them to our console, let's refresh our minds as to how 74HC595 shift registers work.

Do you remember?

These columns are not just about building cool things but also to learn lots of interesting and useful stuff along the way. So even if you have not yet committed to building the games console, you can still join in with the following experiments.

Let's start with the 8-segment LED bar display we introduced in our January 2025 column. The way we left things is depicted in Fig.1. An image of the full breadboard along with all

the other components and wires can be downloaded in file **CB-jun25-brd-01.pdf** as part of the June 2025 download package from the PE website (<https://pemag.au/link/ac54>).

Observe that we are using 560Ω current-limiting resistors with green-blue-brown bands. We are using a 5V supply from the Arduino Uno to power our breadboard and the red LEDs in our display have a forward voltage drop of 2V and a maximum forward current of 20mA.

Using Ohm's law of $V = IR$, and rearranging it to give $I = V/R$, our 560Ω resistors limit the current going through each LED to $(5V - 2V) \div 560\Omega = \sim 5.5mA$. In addition to being bright enough for our purposes here, that is below the maximum of 6mA that 74HC595

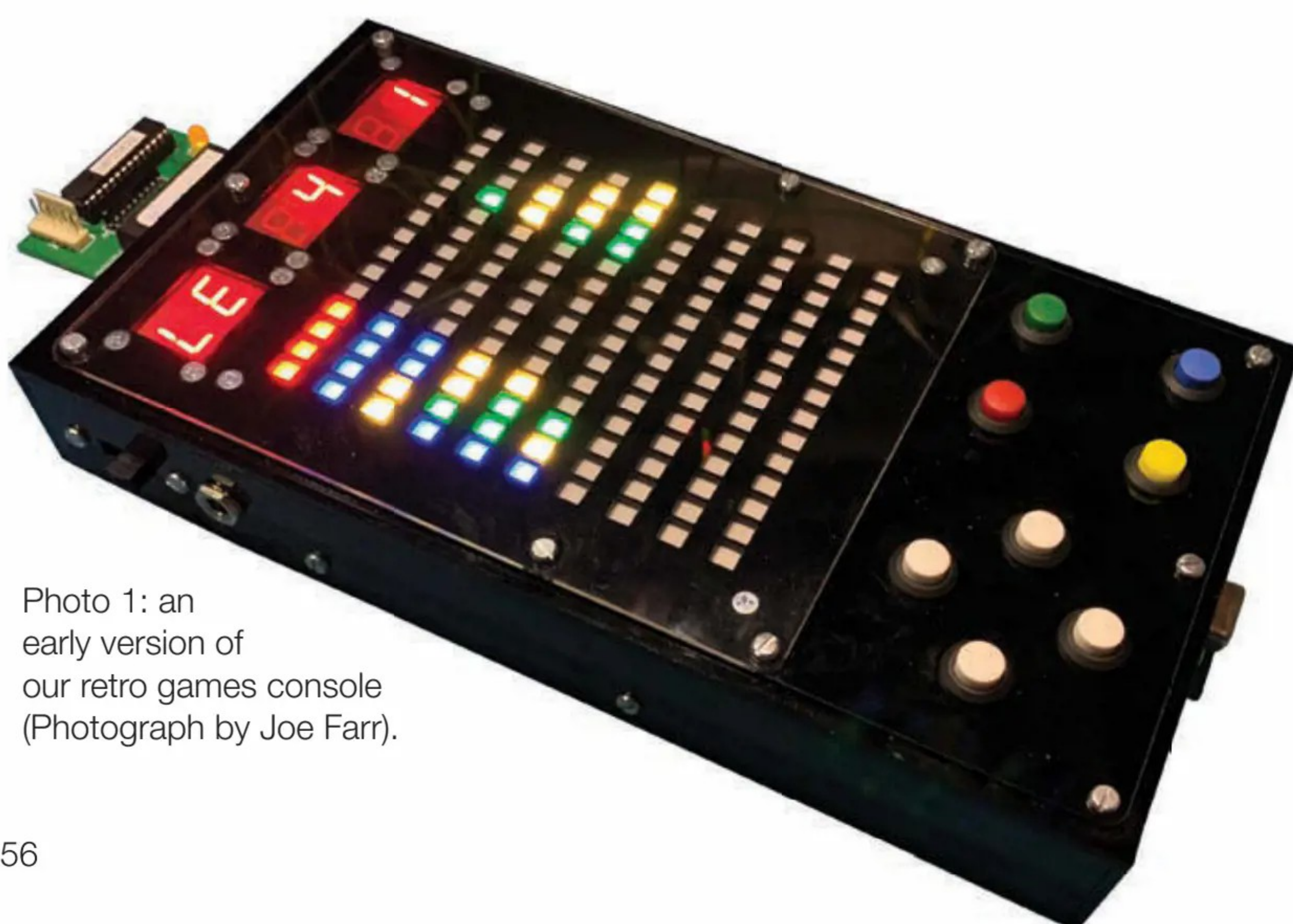


Photo 1: an early version of our retro games console (Photograph by Joe Farr).

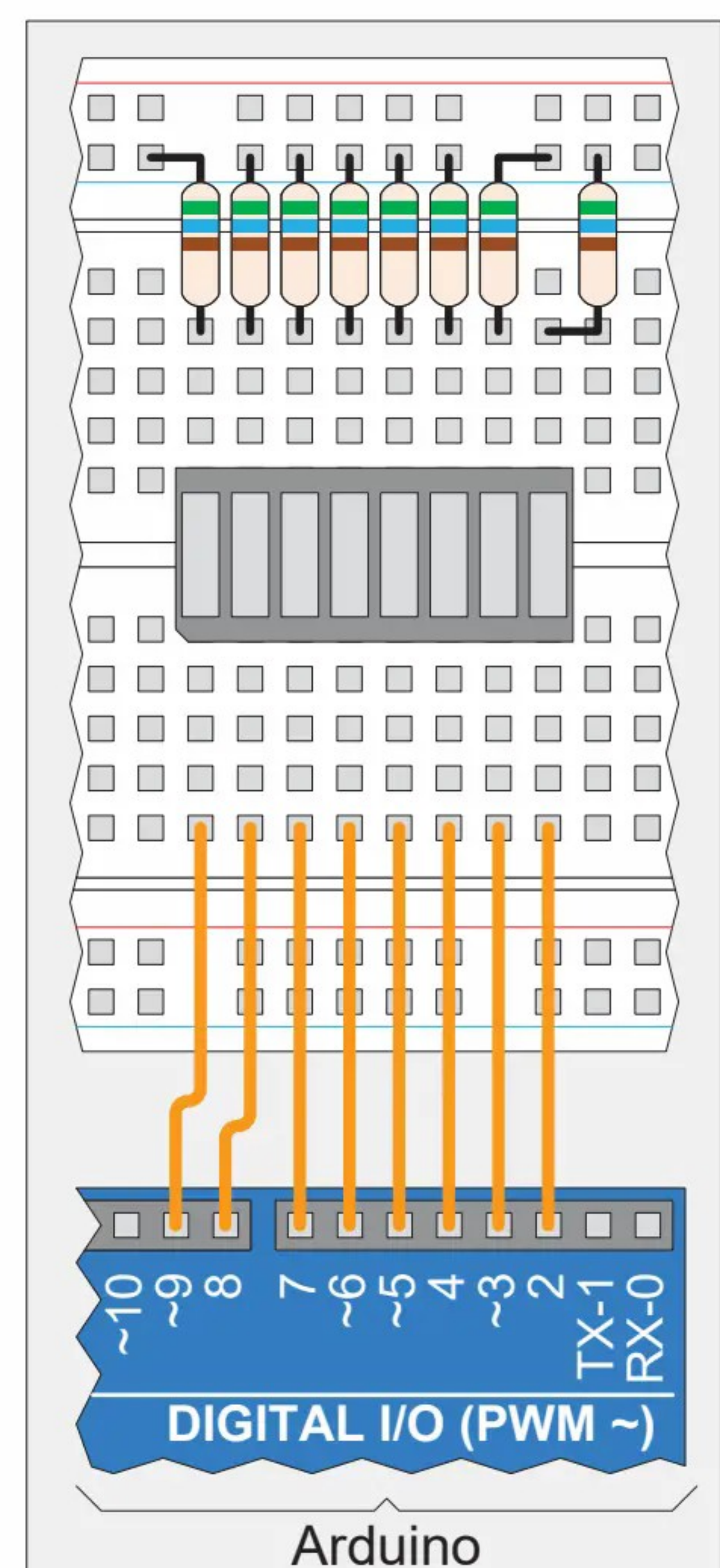


Fig.1: the way we left things last time.


```

12 // Declare pins driving the display
13 int PinsSegs[NUM_SEGS] =
14 {2, 3, 4, 5, 6, 7, 8, 9};

```

Listing 1(a): our pin mapping declarations.

shift registers can sink or source on their outputs.

The way we've connected things in Fig.1, one side of each current-limiting resistor is connected to the breadboard's ground rail. This means a 0 (**LOW**) or 1 (**HIGH**) on the corresponding Arduino pin will switch that display segment off or on, respectively.

As part of the January column, we created a program to generate and display 8-bit random numbers (you can download a copy of this code in file **CB-jun25-code-01.txt**).

To be honest, this program is a tad clunky, but it serves as a springboard for what's to come. The first part of the program that we need to consider is where we declare the pins we're using to drive the display, as illustrated in Listing 1(a). We are using the convention that the listing number (1 in this case) corresponds to the numerical part of the matching code file name ("01" here).

Remember that **NUM_SEGS** has been defined as 8, and the elements in the corresponding 8-element array are referenced as 0 to 7. For example, **PinsSegs[0]** and **PinsSegs[7]** are assigned values of 2 and 9, which are the pins we are using to drive the right-most and left-most segments in our display, respectively.

Now let's consider the **loop()** function illustrated in Listing 1(b). The **for()** loop on line 36 cycles its **iSeg** control variable from 0 to 7. For each pass through the loop, we call the **digitalWrite()** function on line 38. This accepts two arguments: the number of the pin we wish to control and the value we wish to write to that pin.

For the first argument, we are using the **iSeg** value to index into our **PinsSegs[]** array of pin numbers. With respect to the second argument, we are using the Arduino's **random()** function to generate random 0 and 1 values.

As we discussed last month, this function accepts two arguments that we might call **min** and **max**. The **min** value is inclusive, while the **max** value is exclusive, which means the function will generate a random integer between **min** and **(max - 1)**. Since we are using **random(0,2)**, this will return either 0 or 1, which is like flipping a coin in the air and calling "heads" or "tails".

Remember that the Arduino treats 0 and **LOW** as being synonymous, and similarly for 1 and **HIGH**.

Bytes are better

There are many things to dislike about our current implementation,

Listing 1(b): our code for generating random values bit by bit. This is not the most efficient method, but it works!

not least that we are calling the **random()** function eight times to generate each 8-bit value on the display. Obviously, it would be better to call this function only once. That's just what we are going to do, as shown in Listing 2(a), part of the file named **CB-jun25-code-02.txt**.

We've created a **DisplayByte()** function on line 44. This function has one parameter, an 8-bit unsigned integer called **thisByte** that will contain the 8-bit value we wish to display.

Once again, we have a **for()** loop on line 46 that cycles its **iSeg** control variable from 0 to 7. For each pass through the loop, on line 48, we use a bitwise AND (&) to mask out the least-significant bit of **thisByte** and test to see if it's 1. Then we use the result from this test to switch the corresponding segment on or off.

For more details, see the *Masking and the C/C++ Bitwise Operators* column on my website (<https://pemag.au/link/ac1q>).

The last thing we do in the loop, on line 53, is to shift the value in **thisByte** one bit to the right, thereby preparing it for the next pass through the loop.

In the case of the main **loop()** function, on line 38 we use a single call to the Arduino's **random()** function to generate a random number between 0 and 255, and we assign that value to an 8-bit unsigned integer called **tmpVal**.

On Line 39, we pass this value into our **DisplayByte()** function to be displayed. Later, on line 40, we pause for a moment to give ourselves time to observe our new random value on the 8-segment display.

Count on me

The way we've re-architected our program—in particular, the addition of our **DisplayByte()** function—allows us to do all sorts of interesting things, like easily displaying a binary

```

33 // Do this over and over again
34 void loop ()
35 {
36     for (int iSeg = 0; iSeg < NUM_SEGS; iSeg++)
37     {
38         digitalWrite(PinsSegs[iSeg], random(0,2));
39     }
40     delay(CYCLE_TIME);
41 }

```

```

33 // Do this over and over again
34 void loop ()
35 {
36     uint8_t tmpVal;
37
38     tmpVal = random(0, 256);
39     DisplayByte(tmpVal);
40     delay(CYCLE_TIME);
41 }
42
43 void DisplayByte (uint8_t thisByte)
44 {
45     for (int iSeg = 0; iSeg < NUM_SEGS; iSeg++)
46     {
47         if ( (thisByte & 1) == 1)
48             digitalWrite(PinsSegs[iSeg], SEG_ON);
49         else
50             digitalWrite(PinsSegs[iSeg], SEG_OFF);
51         thisByte = thisByte >> 1;
52     }
53 }
54
55 }

```

Listing 2(a): generating random values by the byte instead.

```

33 // Do this over and over again
34 void loop ()
35 {
36     uint8_t tmpVal = 0;
37
38     while(1)
39     {
40         DisplayByte(tmpVal);
41         tmpVal = tmpVal + 1;
42         delay(CYCLE_TIME);
43     }
44 }

```

Listing 3(a): displaying a binary count.

count sequence, for example. Consider Listing 3(a), part of the file named **CB-jun25-code-03.txt**.

All we've done is tweak the main **loop()** function. On line 36, we declare an 8-bit unsigned integer called **tmpVal** and initialise it to 0. Then we use the **while()** loop on line 38 to cycle around calling our **DisplayByte()** function to display the value in **tmpVal**, after which we increment this value.

Since **tmpVal** is an 8-bit field, it can only contain values between 0 and 255, after which the next count will return it to 0.

The **while(1)** statement is just a cunning way to keep on cycling around this loop without ever exiting the main **loop()** function. Remember that the

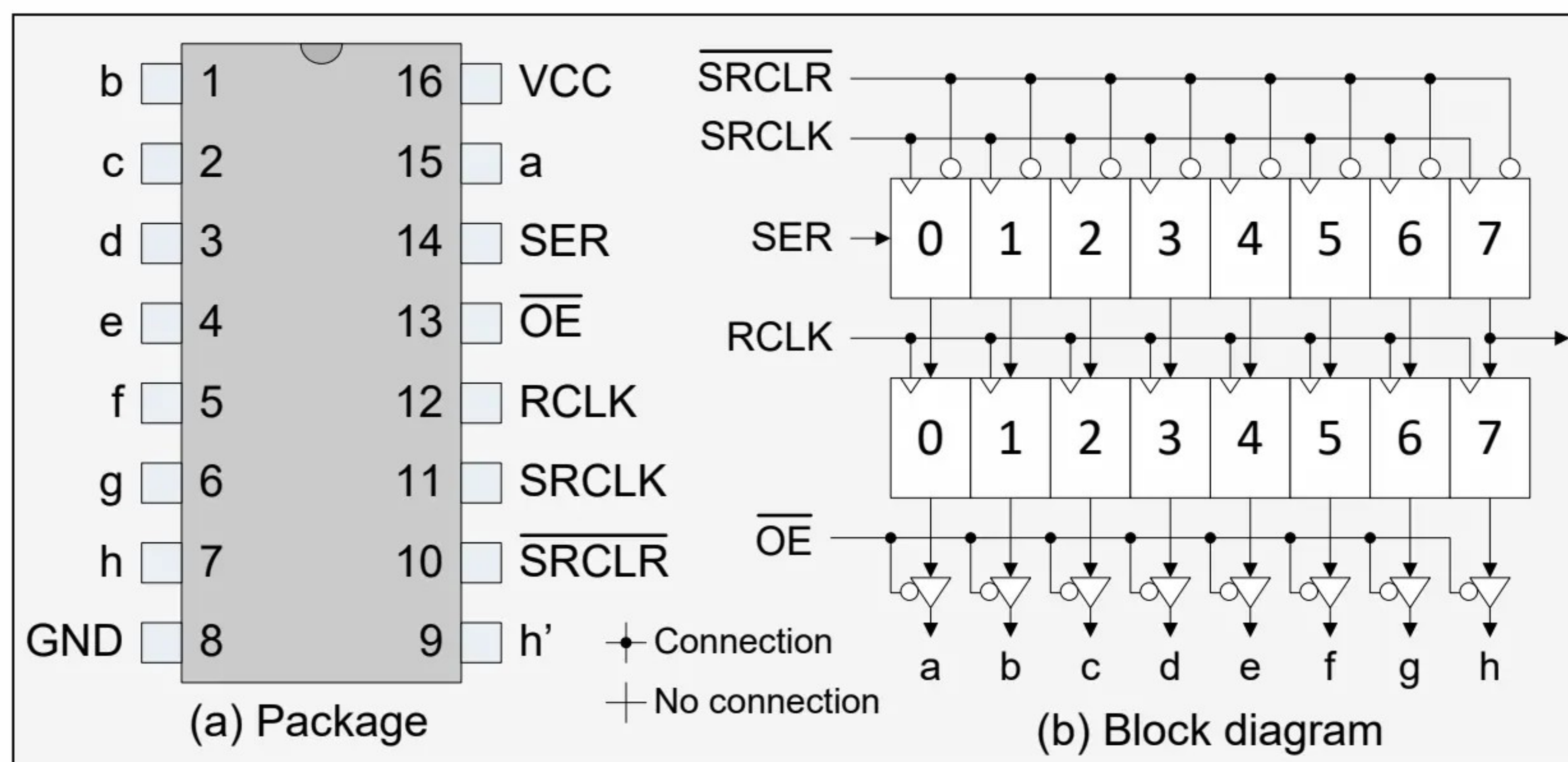


Fig.2: the pinout and internal gates of a 74HC595 serial-to-parallel shift register IC.

Arduino treats 1 and **true** as being synonymous; similarly for 0 and **false**. Since **true** is always true, the while loop never ends!

Shift over

This is where things start to get super exciting (said Max, super excitedly). As mentioned earlier, we are going to use a 74HC595 integrated circuit (IC), an 8-bit shift register, to drive our 8-segment LED display.

This component is presented in a 16-pin package, as per Fig.2(a). It contains two 8-bit registers that operate independently of each other – see Fig.2(b). We can think of the upper register as being the shift register itself, and we can view the lower register as being the output register.

A detailed description of this device's operation is provided in its data sheet (<https://pemag.au/link/ac10>), but a quick overview follows. Let's start with the **SRCLR** ('shift register clear') input. The bar over its name indicates that this signal is active-low. When **SRCLR** is placed in its active (0)

state, it clears all the bits in the shift register to 0.

Once the **SRCLR** input has been returned to its inactive (1) state, a rising edge (a 0 to 1 transition) on the **SRCLK** ('shift register clock') input will load the 0 or 1 value presented to the **SER** ('serial data') input into bit 0 of the shift register. At the same time, the original contents of bit 0 will be loaded into bit 1, the original contents of bit 1 will be loaded into bit 2, and so on down the line.

One way to visualise this is as a bucket brigade (a chain of people acting to put out a fire by passing buckets of water from hand to hand).

If the outputs from the shift register were used to drive something like a 7-segment numerical display directly, that display would flicker as the new values were loaded into the register. This explains why we also have an output register.

It's only when we are ready to present the contents of the shift register to the outside world that we apply a rising edge to the **RCLK** ('output register clock') input, thereby loading the current contents of the shift register into the output register.

The only remaining control signal to consider is the active-low **OE** ('output enable') input. When this input is in its active (0) state, the contents of the output register will be made available to the outside world. By contrast, when this input is in its inactive (1) state, the outputs will be effectively disconnected from the outside world (in our case, we will set it so the outputs are always enabled).

Adding the 74HC595

A schematic showing the way we are going to intro-

duce the shift register into our circuit is presented in Fig.3; the *1, *2, *3, and *4 annotations refer to notes, not pin numbers. For this first test, we are connecting the **SER** ('serial in') input to logic 1 via a 10kΩ pull-up resistor. Also, we are connecting the **OE** ('output enable') input to logic 0 via a 10kΩ pull-down resistor.

We will employ three of the breadboard-friendly momentary pushbutton switches we used in our *Arduino Bootcamp* columns. By 'momentary' we mean these are only closed while they are being pressed; they immediately return to their inactive (open) states when released.

The **SRCLK** input is connected to 0V via a 10kΩ **pull-down** resistor. It's also connected to one side of one of our pushbutton switches, the other side of which is connected to 5V. When this switch is pressed, the resulting direct connection to 5V will overwhelm the 10kΩ pull-down resistor to 0V, which means the **SRCLK** input will be presented with a 0 to 1 transition (a 'rising edge'). The **RCLK** input is connected in the same way.

By comparison, the active-low **SRCLR** input is connected to 5V via a 10kΩ **pull-up** resistor. In this case, the other side of the associated switch is connected to 0V. When this switch is pressed, the resulting direct connection to 0V will overwhelm the 10kΩ pull-up resistor to 5V, which means the **SRCLR** input will be presented with its active (0) state.

While a range of values would work, we chose the 10kΩ values for our pull-up and pull-down resistors to suit the 74HC IC technology we're working with. Other chip types might need different values; see *How to Handle the Inputs to Unused Logic Gates* column for more details (<https://pemag.au/link/ac55>).

Updating the breadboard

Unplug the USB cable from your Arduino to power everything down. Remove the wires connecting Arduino pins 2 through 9 to the breadboard, but leave the wires connecting 0V and 5V on the Arduino to the 0V and 5V rails on the breadboard.

Add the 74HC595 to the breadboard as illustrated in Fig.4 (an image of the full breadboard is available in the downloadable file named **CB-jun25-brd-02.pdf**). Observe that the notched end of the package is shown on the left, which means pin 1 is in the lower left corner and pin 16 is in the upper left corner.

Although this is the opposite of the way we did things with our *Arduino Bootcamp* clock, it's the way chips are

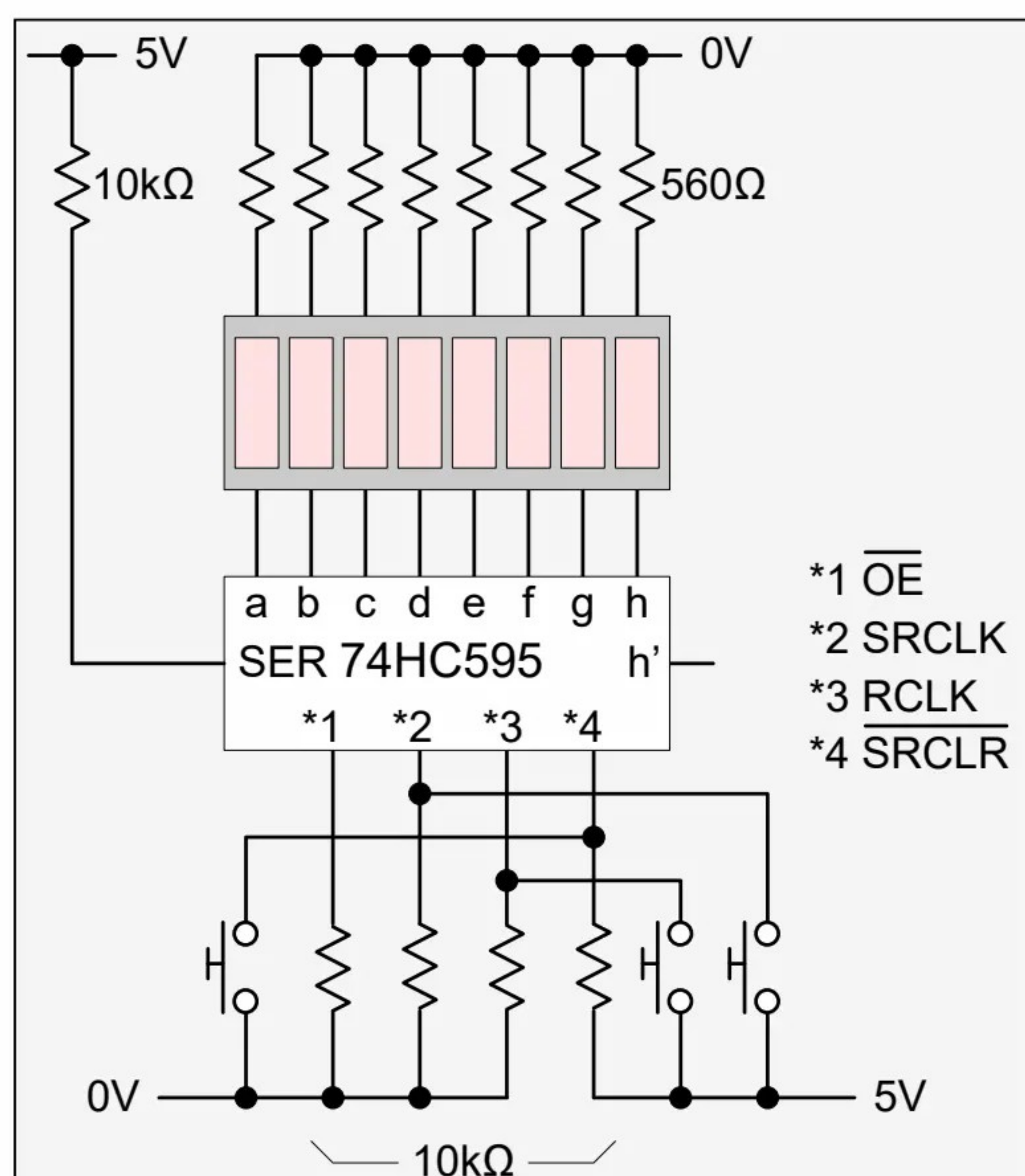


Fig.3: this circuit uses buttons to control the 74HC595.

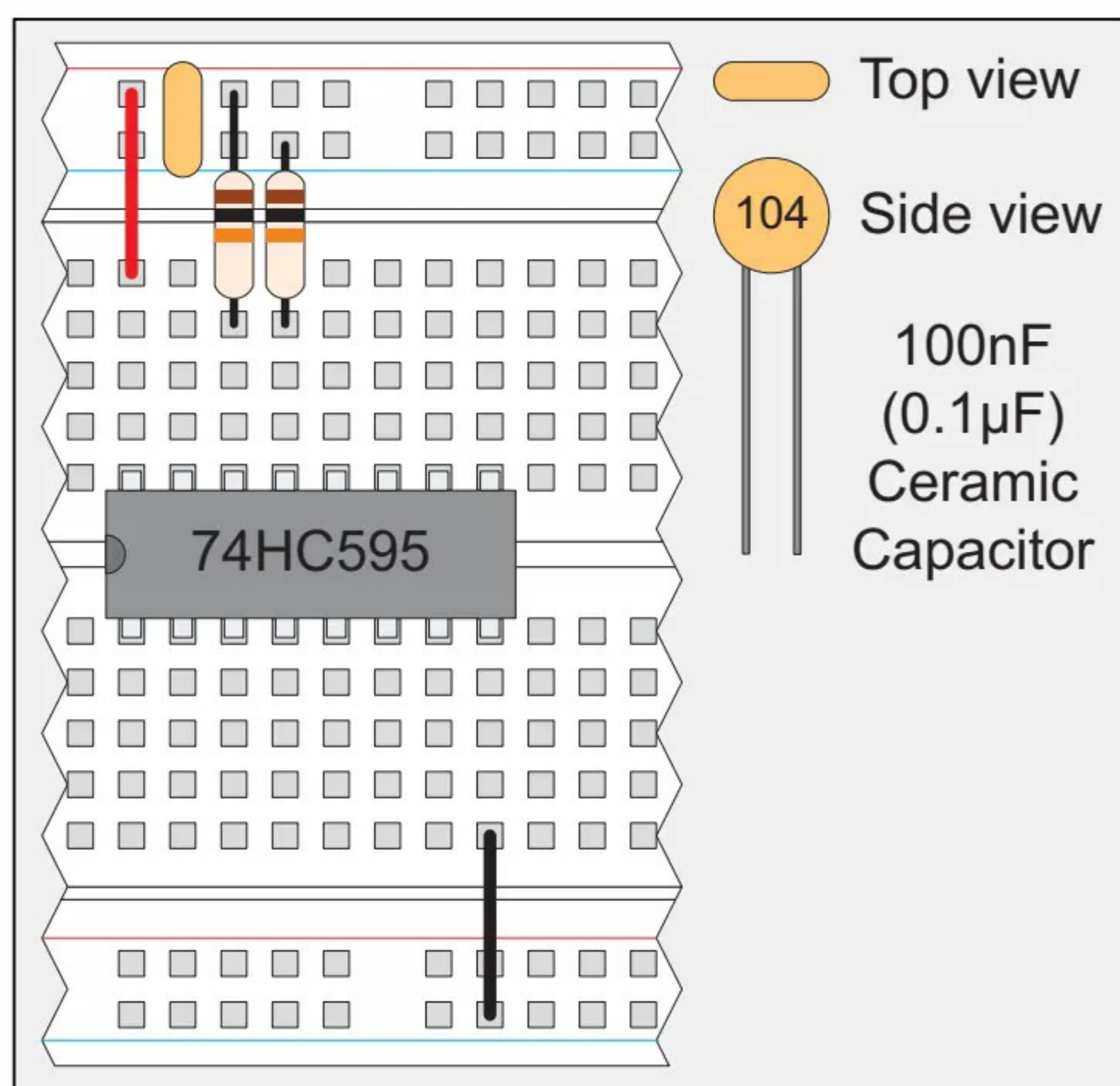


Fig.4: adding the shift register to our breadboard.

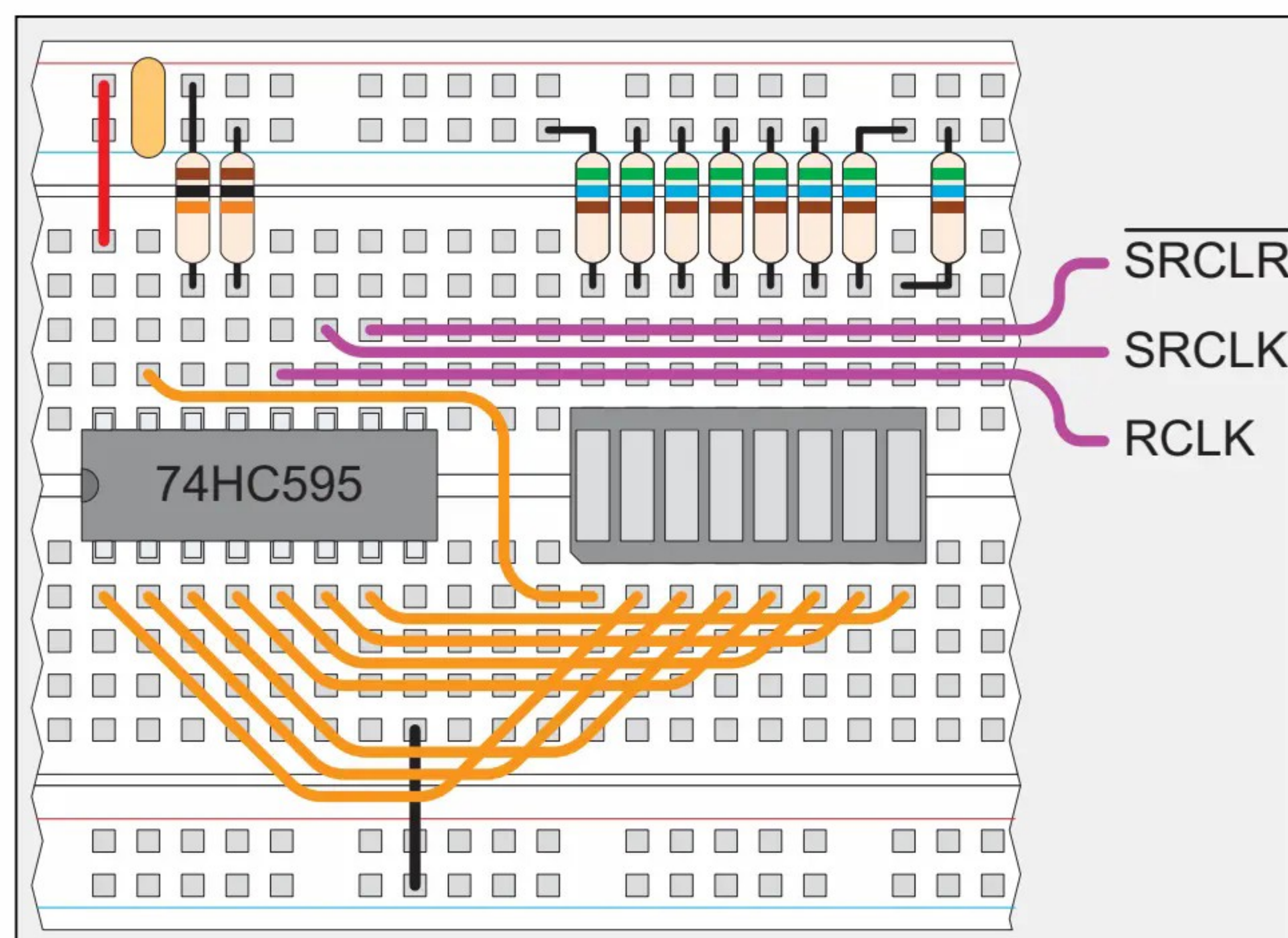


Fig.5: connecting the shift register IC to the 8-segment LED array.

usually orientated on a breadboard or printed circuit board (PCB).

Make sure to add the red wire connecting pin 16 (VCC) to the 5V rail and the black wire connecting pin 8 (GND) to the 0V rail. Also add the 10kΩ pull-up resistor between pin 14 (the **SER** input) and the 5V rail, and the 10kΩ pull-down resistor between pin 13 (the **OE** input) and the 0V rail.

While you're at it, add a decoupling/bypass ceramic capacitor as shown. As we discussed in the August 2024 column, these components aren't polarised, so we can connect them either way round. A good rule of thumb is one 100nF (0.1µF) capacitor per IC. This should be positioned between the 5V and 0V rails as close to the red wire as possible.

The "104" annotation on the capacitor represents $10 \times 10^4 = 100,000$, measured in picofarads (pF), which equates to 100nF.

Reconnect the USB cable to power everything up, then use your multimeter to verify that you see 0V on pins 8 and 13, and 5V (give or take) on pins 16 and 14.

Power everything down again, then connect the outputs from the shift register to the inputs of the 8-segment display as illustrated in Fig.5. Observe the three wires connected to the **RCLK**, **SRCLK** and **SRCLR** inputs.

In the fullness of time, we will be driving these signals from our Arduino. First, however, we are going to add the pushbutton switches from Fig.3 as illustrated in Fig.6 (the full breadboard can be seen in the file named **CB-jun25-brd-03.pdf**).

These switches are available in 2-pin or 4-pin packages. Whichever version you are using, make sure the pins are presented as shown in Fig.6. With the 4-pin versions, you can use the continuity setting on your

multimeter (or measure the resistance) to determine which pin pairs are connected.

It's important to deploy the switches with the correct orientation. In the case of a 2-pin version, if you rotate it by 90°, its associated wires will remain forever unconnected, irrespective of how (un)enthusiastically you press the button. By comparison, if a 4-pin switch is rotated by 90°, its associated wires will be permanently connected.

Neither condition will cause any damage, but the switches won't operate as expected.

Missing a cunning trick

I just remembered a cunning trick my friend Joe Farr taught me with respect to pull-up and pull-down resistors. For 74HC-series components (with which we use 10kΩ to 100kΩ values), Joe uses 47kΩ (yellow-violet-orange) for his pull-ups and 22kΩ (red-red-orange) for his pull-downs.

This way, when Joe is in the process of debugging a circuit, just seeing a yellow-violet or red-red colour band combo tells him that he's looking at a pull-up or pull-down resistor, respectively. I would have used this trick

myself if I'd remembered it, but I didn't, and I'm not going back to modify all my diagrams now (sorry).

Let's run some tests

Our first test is to simply power everything up and observe what happens. In my case, this was... absolutely nothing. I powered things off and on again several times... still nothing.

To be honest, this was a bit of a surprise, and it presented a bit of a poser, because the 74HC595 isn't supposed to contain a special power-on-reset (PoR) circuit to clear its internal registers and initialise them with values of 0. Thus, I was expecting to see a random collection of on and off LED segments that would be different every time power was removed and reapplied.

Currently, I have no explanation for why I'm not seeing random values. If you have any ideas, or if your register behaves differently than mine, please feel free to drop me a line to share your thoughts and experiences.

Just to be on the safe side, press and release the **SRCLR** (right-hand) switch and then press and release the **RCLK** (left-hand) switch. This will

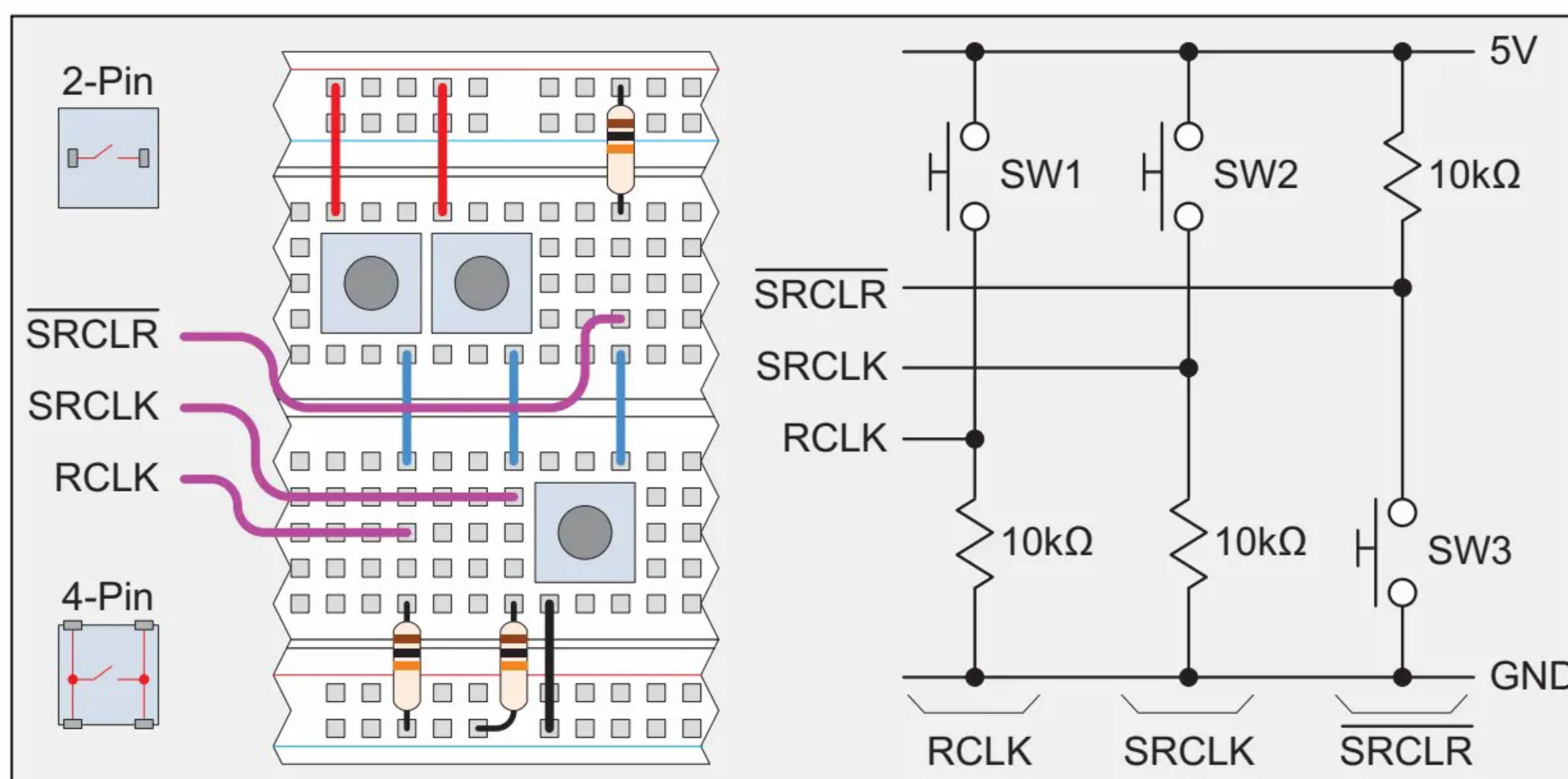


Fig.6: adding and connecting the switches to our existing circuit on the breadboard.

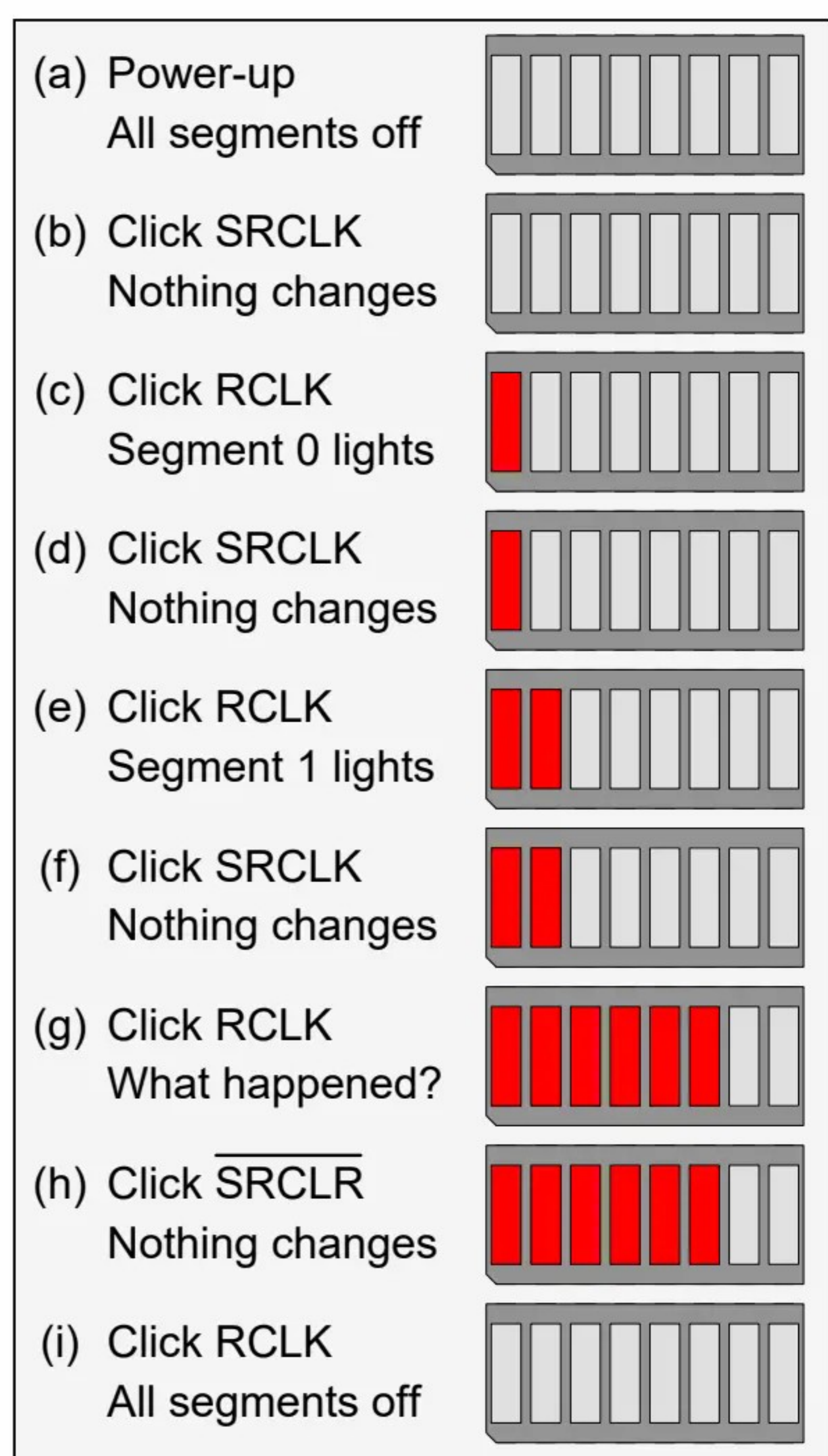


Fig.7: testing the shift register.

ensure that all our segments are off, as illustrated in Fig.7(a).

Remember that we've permanently connected the **SER** ('serial in') input to logic 1 (5V). Press and release the **SRCLK** (middle) switch to load this 1 into bit 0 of the shift register. Observe that nothing happens on the display as illustrated in Fig.7(b). Now press and release the **RCLK** (left-hand) switch. This copies the contents of the shift register into the output register.

What we are expecting to see is only the first segment light up as illustrated in Fig.7(c). Just for giggles and grins, let's assume that this is what actually happens.

Press and release the **SRCLK** switch

again. Observe that nothing happens on the display, as illustrated in Fig.7(d). Now press and release the **RCLK** switch again. We expect to see two segments active as illustrated in Fig.7(e). Once again, let's assume that this is what happens.

Let's go for broke. Press and release the **SRCLK** switch one more time. Nothing happens, as shown in Fig.7(f). Press and release the **RCLK** switch one more time. Let's assume that, instead of the first three segments being active, the first six light up, as seen in Fig.7(g). What just happened?

I'll tell you in a moment. First, however, press and release the **SRCLR** switch. This clears the internal shift register, but not the output register, so nothing changes, as illustrated in Fig.7(h). Finally, press and release the **RCLK** switch to copy the contents of the internal shift register into the output register, and watch all the segments turn off as shown in Fig.7(i).

Perform some more experiments on your own. It may be that your circuit appears to work as expected most of the time (I hope not, because that would diminish the impact of what we are about to discuss).

Alternatively, you might consistently see multiple segments light up. Or you could fall in the middle, with extra segments lighting up occasionally. My own circuit seems to happily meander between all three scenarios. So, WTW (what the what) is going on?

Bouncy, bouncy!

The problem is that switches bounce. That's just what they do. When we flick a toggle switch or press a pushbutton switch, we assume it transitions cleanly from one state to another; that is, from on to off or vice versa.

However, due to factors like spring action and mechanical inertia, a switch does not settle immediately. Instead,

it suffers a series of rapid, unintended on-and-off connections (or 'bounces') before stabilising. We can visualise this as dropping a ping-pong ball onto a hard floor.

Photo 2 shows waveforms captured on my oscilloscope, reflecting three switches being activated. From top to bottom, we see a toggle switch, a push-button switch and a limit (micro) switch.

The 'exciting' thing is that these waveforms can change dramatically between multiple switches of the same type, or even between the same switch being activated and deactivated multiple times. Also, a switch's bounce characteristics can vary as a function of temperature, humidity, and age... to name but a few factors.

To put this into perspective, we're having problems controlling a simple 8-segment LED display. Suppose our pushbutton switch was being used to control a missile launch system. Do I need to say more?

Arduino to the rescue!

I discuss the concept of switch bounce and ways to mitigate it in ~~excruciating~~ *excruciating* enthralling detail in my *Ultimate Guide to Switch Debounce* columns (<https://pemag.au/link/ac56>). We can debounce signals from switches using both hardware and software techniques. In this case, we're going to use our Arduino to debounce the signals in software.

We'll start by powering everything down. Next, we'll break the direct connections between the switches and our shift register, then we'll insert the Arduino in the middle as depicted in Fig.8 (an image of the full breadboard can be found in the file named **CB-jun25-brd-04.pdf**).

We can summarise the switch bounce situation for our **SRCLK** signal in Fig.9 (the waveforms will be similar for the **RCLK** signal and inverted for the **SRCLR** signal). The signal coming from the switch (SW) is normally 0V. It transitions to 5V when its associated button is pressed.

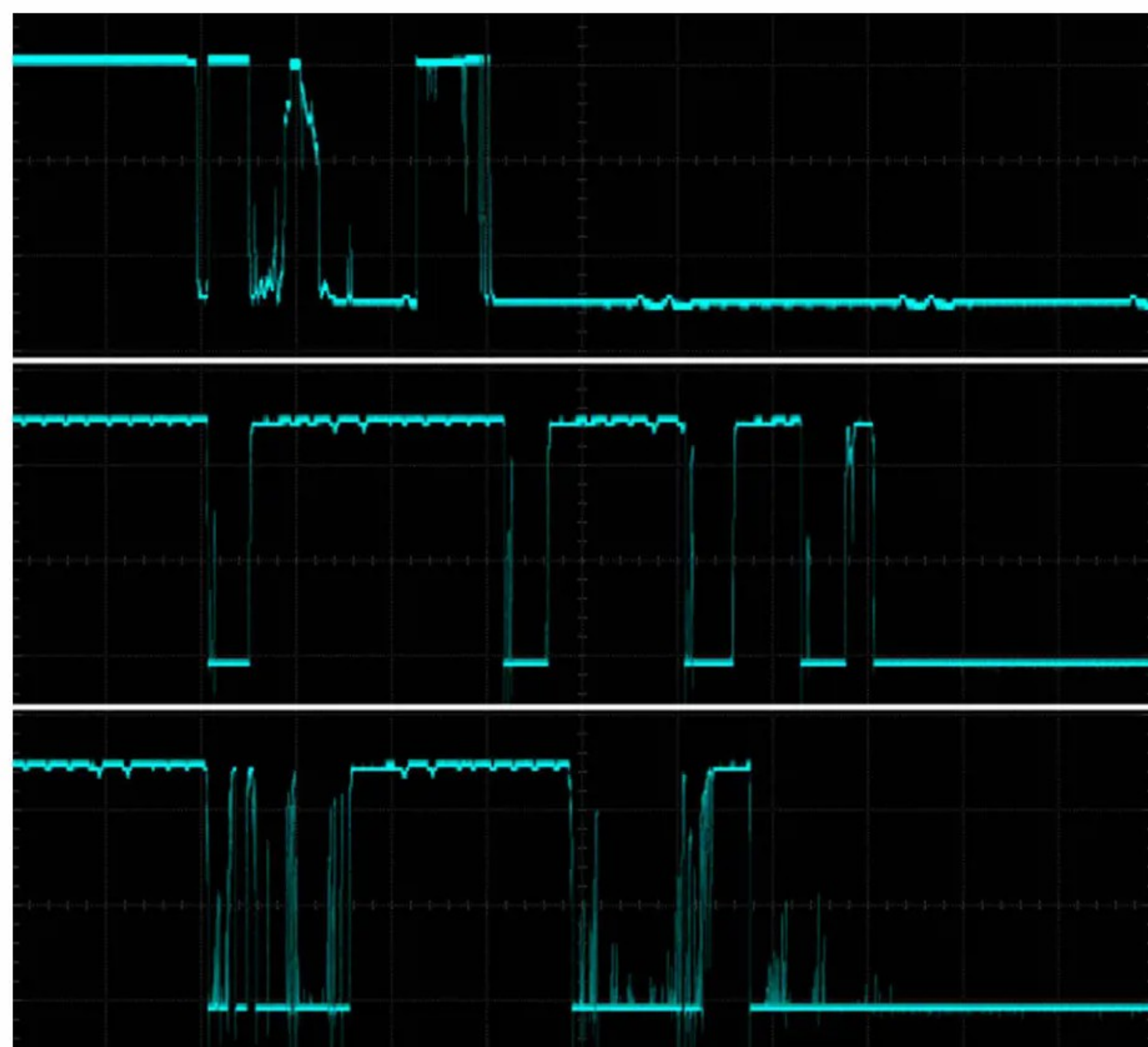


Photo 2: waveforms from three switches being actuated.

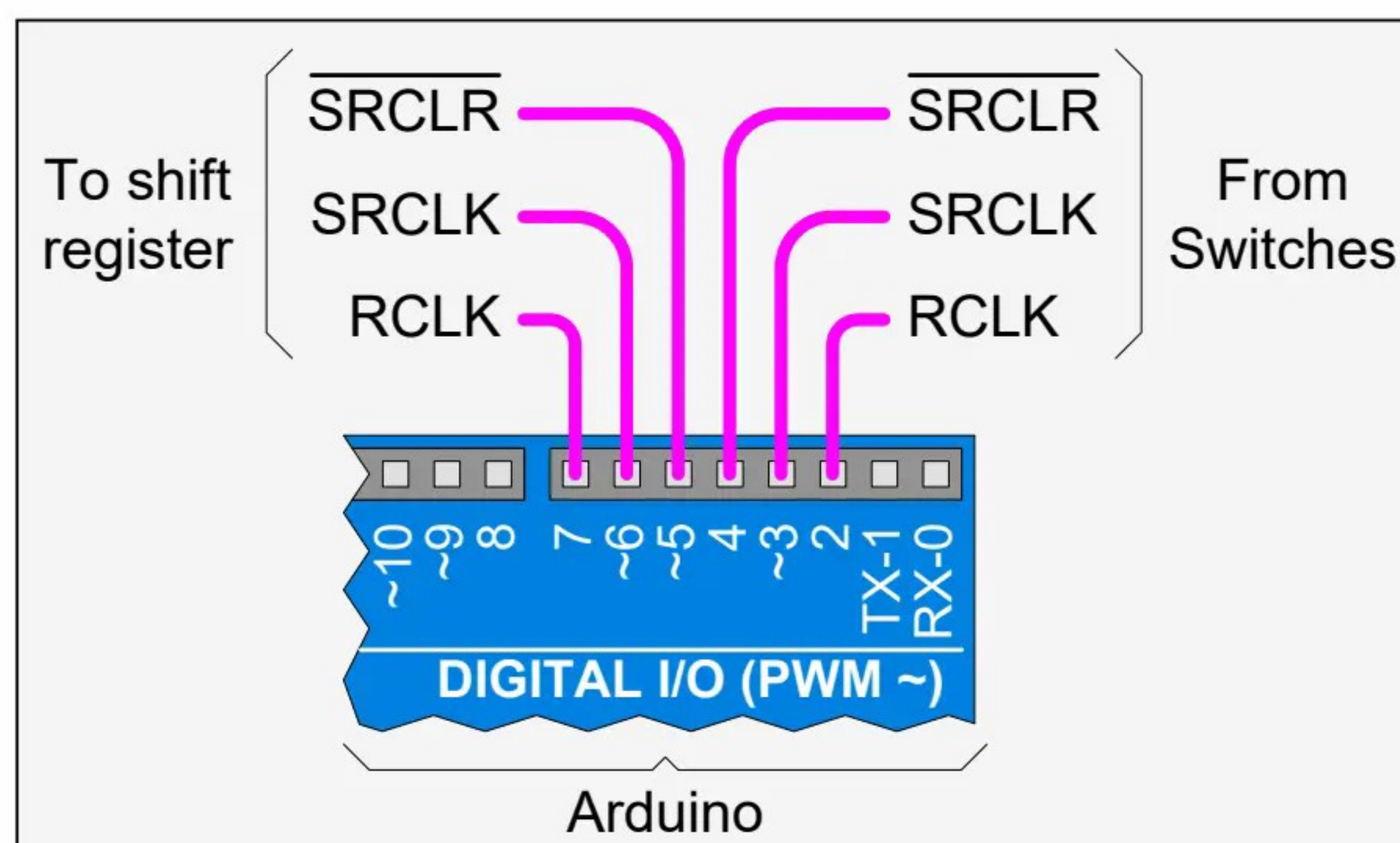


Fig.8: inserting the Arduino into the switch signal paths.

Observe that bouncing can occur both when the switch is activated (pressed) and deactivated (released). For the purposes of our discussions here, let's assume that our switches can bounce anywhere from 1 to 100 times over a period not exceeding six milliseconds (6ms).

In an ideal world, starting from the switch being in its inactive state, the Arduino would detect the leading edge of the activation and cause its output to the shift register (SR) to respond immediately. In fact, this is what we are going to do now. However, in our next column, we will discover why this approach may not be quite as ideal as we might suppose.

As seen in Fig.9, we now have two options. The first is for the output from the Arduino to follow the input from the switch (but without any bounce, of course). The second is for the output from the Arduino to present a pulse to the shift register when the switch is activated, and to subsequently do nothing until the next activation of the switch. Let's consider these options in turn.

Software debounce option #1

I have to say that I'm ashamed about what I'm about to show you. This is one of the simplest ways to debounce our switches, but it's 'quick and dirty'; it lacks elegance, and it's the opposite of robust. Still and all, as they say, it will do the job we need to do here, and it will provide something to compare ourselves against in the future.

I just created a rudimentary program (in the file named **CB-jun25-code-04.txt**). Earlier, I said that we would assume our switch bounce would not exceed 6ms. Based on this, we've defined a delay value called **JUST_A_BIT** as being 10ms.

Next, we specify the pins we're using on the Arduino to match those shown in Fig.8. In the **setup()** function, we say we want to use pins 2, 3 & 4 as inputs, and pins 5, 6 & 7 as outputs. We also set the outputs to their inactive states.

```
30  if (digitalRead(PinSrclkFromSw) == HIGH)
31  {
32      digitalWrite(PinSrclkToSr, HIGH);
33      delay(JUST_A_BIT);
34
35      while (digitalRead(PinSrclkFromSw) == HIGH)
36      {
37      }
38      digitalWrite(PinSrclkToSr, LOW);
39      delay(JUST_A_BIT);
40  }
41
42
```

Listing 4(a): software debounce option #1.



Fig.9: summarising the switch bounce situation for the SRCLK signal.

In the **loop()** function, we process each switch in turn. Consider the way we handle the **SRCLK** signal, as illustrated in Listing 4(a). Remember that the **SRCLK** signal from the switch is usually inactive (**LOW**). Thus, we are looking for the leading edge when it first transitions to active (**HIGH**).

So, every time we go around the loop, on line 30, we read the pin connected to the switch and check to see if this signal has gone **HIGH**, meaning the switch has been pressed. If not, we continue to the next switch.

When we see that the switch has been pressed, we use line 32 to place a **HIGH** value on the **SRCLK** signal to the shift register. Then, on line 33, we wait for **JUST_A_BIT** (10ms) to ensure that the switch has stopped bouncing.

Some users will press and release the switch rapidly, while others may take a more leisurely approach. Either way, on line 35, we use a **while()** loop to wait for the switch to be released (think of this as "while the switch is still active, don't do anything (or do nothing)").

When we see that the switch has been released, we use line 38 to place a **LOW** value on the **SRCLK** signal to the shift register. Since the switch may bounce when it's released, on line 39, we once again wait for **JUST_A_BIT** to ensure that the switch has stopped bouncing.

Load this program into your Arduino and verify that everything functions as expected.

Software debounce option #2

I just made a small modification to our program, in the file named **CB-jun25-code-05.txt**. As you can see in Listing 5(a), this is very similar to option #1. All we've done is to move the statement that places a **LOW** on the **SRCLK** signal to the shift register from line 38 in Listing 4(a) to line 33 in Listing 5(a).

Now, as soon as the switch is activated, the Arduino presents a positive-going pulse on the **SRCLK** signal feeding the shift register. The Arduino then pauses to let any bouncing stop, waits for the switch to be released, pauses again to let any bouncing stop, and then proceeds to look at the other switches.

Load this new version of our program into your Arduino and verify that everything continues to function as we expect.

Clear the display again, then press the **SRCLK** switch, followed by the **RCLK** switch six times. Once again, this results in the first six segments being lit.

Software debounce option #3

I know... Fig.9 showed only two options. Still, I've been saving the best for last!

Earlier, we noted that one reason for our 74HC595 chip having both a shift register and an output register is to prevent the annoying flicker that would occur if we were using the shift register to drive something like a 7-segment numerical display directly.

To illustrate this in action, use the **SRCLR** and **RCLK** switches to clear the

```
30  if (digitalRead(PinSrclkFromSw) == HIGH)
31  {
32      digitalWrite(PinSrclkToSr, HIGH);
33      digitalWrite(PinSrclkToSr, LOW);
34      delay(JUST_A_BIT);
35
36      while (digitalRead(PinSrclkFromSw) == HIGH)
37      {
38      }
39      delay(JUST_A_BIT);
40  }
41
42
```

Listing 5(a): software debounce option #2.

contents of the register and display. Next, press and release the **SRCLK** switch six times (nothing changes on the display), then press and release the **RCLK** switch (the first six segments light simultaneously).

This separation of registers is great when we need it, but it's an encumbrance for what we are currently doing. For example, use the **SRCLR** and **RCLK** switches to clear the contents of the register and display.

Now press the **SRCLK** switch, followed by the **RCLK** switch six times. Once again, this results in the first six segments being lit. I don't know about you, but my poor finger is becoming fatigued by constantly having to press the **RCLK** switch after pressing the **SRCLK** or **SRCLR** switches.

Happily, one great thing about doing things in software is that we can use one stimulus signal to trigger multiple responses. Suppose, for example, we modify the actions associated with our **SRCLK** and **SRCLR** signals, as illustrated in Fig.10.

In this case, when we see a rising edge on the signal from the **SRCLK** switch, we can generate a positive-going pulse on the **SRCLK** signal to the shift register followed by a positive-going pulse on the **RCLK** signal to the shift register.



Similarly, when we see a falling edge on the signal from the **SRCLR** switch, we can generate a negative-going pulse on the **SRCLR** signal to the shift register, followed by a positive-going pulse on the **RCLK** signal to the shift register.

I just made the required modifications to our program, in the file named **CB-jun25-code-06.txt**. As we see in Listing 6(a), this is very similar to

option #2. All we've done is to add lines 34 and 35, which generate a positive-going pulse on the **RCLK** signal to the shift register.

If you look at the whole program, you'll see that we've also made appropriate modifications to the code that handles the signal from the **SRCLR** switch, and we've deleted the part that handles the signal from the **RCLK** switch because we no longer need it (which means we could remove this switch from our breadboard if we wished).

Load this new version of our program into your Arduino. Now, when we press the **SRCLK** or **SRCLR** switches, the corresponding actions appear on the display without our having to press the **RCLK** switch. I don't know about you, but this makes me a much happier person.

What's the point?

We've just spent a lot of time pondering switch bounce, but what does that have to do with the price of tea in China (as my dear old grannie used to say)? You must admit that you are unlikely to find tasty tidbits of trivia like this in lesser electronics magazines.

Well, may I invite you to take another look at Photo 1. What do we see below the main LED display? "Good golly, Miss Molly!", I hear you cry, "There are eight juicy pushbutton switches that are going to need to be debounced to enhance our game-playing experience". It's almost as if we had a plan!

Next time

I have so many ideas bouncing my poor old noggin that I barely know where to start. One thing we are going

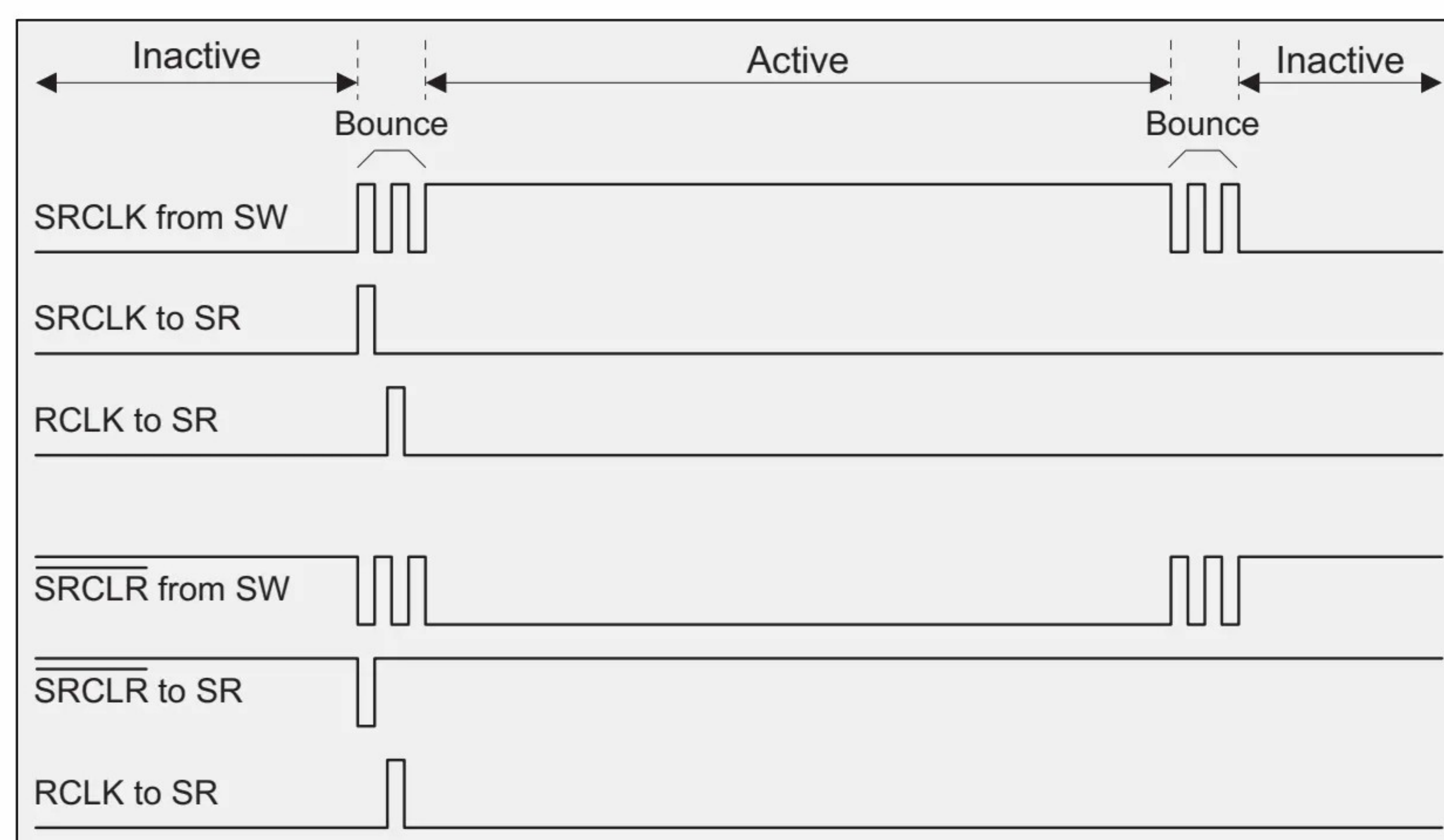


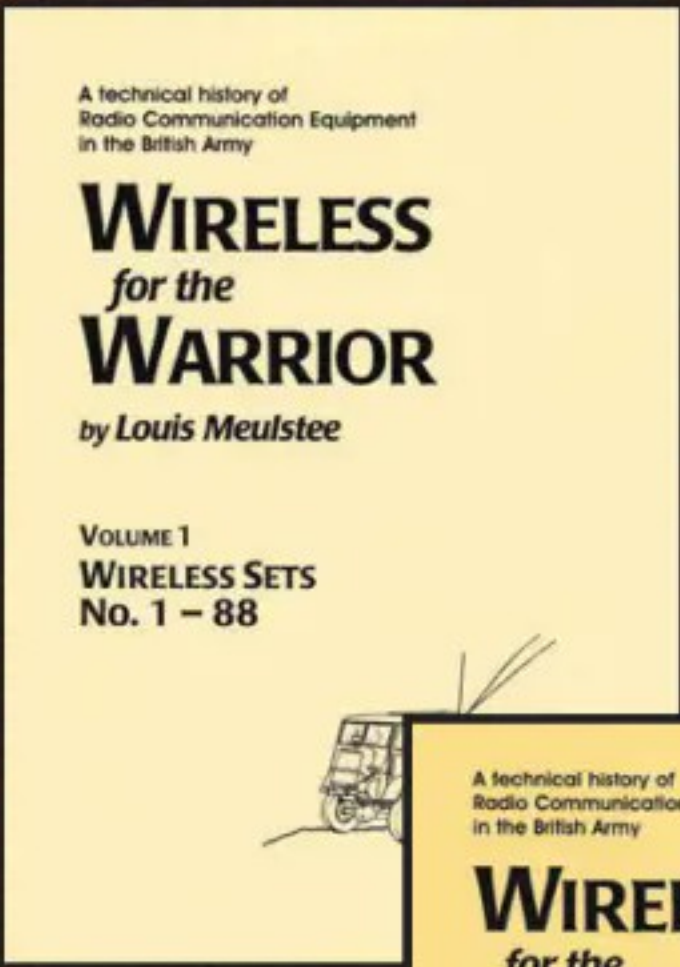
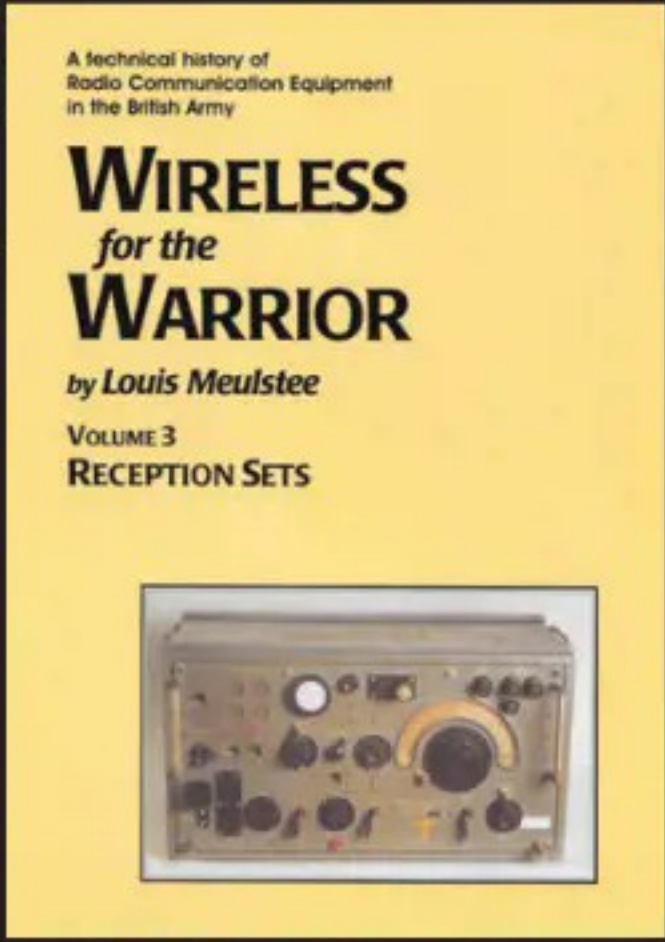
Fig.10: we can modify our software so that one stimulus can trigger multiple responses.

```

30  if (digitalRead(PinSrcLkFromSw) == HIGH)
31  {
32      digitalWrite(PinSrcLkToSr, HIGH);
33      digitalWrite(PinSrcLkToSr, LOW);
34      digitalWrite(PinRclLkToSr, HIGH);
35      digitalWrite(PinRclLkToSr, LOW);
36      delay(JUST_A_BIT);
37
38      while (digitalRead(PinSrcLkFromSw) == HIGH)
39      {
40      }
41      delay(JUST_A_BIT);
42  }

```

Listing 6(a): software debounce option #3.

The Wireless for the Warrior books are references for the history and development of radio communication equipment used by the British Army from the very early days of wireless up to the 1960s.

Volumes 1 & 3 are still available. Order a printed copy now from:

<https://pemag.au/link/ac20>

Useful Bits and Pieces from Our Arduino Bootcamp series

Arduino Uno R3 microcontroller module
 Solderless breadboard
 8-inch (20cm) jumper wires (male-to-male)
 Long-tailed 0.1-inch (2.54mm) pitch header pins
 LEDs (assorted colours)
 Resistors (assorted values)
 Ceramic capacitors (assorted values)
 16V 100µF electrolytic capacitors
 Momentary pushbutton switches
 Kit of popular SN74LS00 chips
 74HC595 8-bit shift registers

<https://pemag.au/link/ac2g>
<https://amzn.to/3O2L3e8>
<https://amzn.to/3O4hnXk>
<https://pemag.au/link/ac2h>
<https://amzn.to/3E7VAQE>
<https://amzn.to/3O4RvBt>
<https://pemag.au/link/ac2i>
<https://pemag.au/link/ac2j>
<https://amzn.to/3Tk7Q87>
<https://pemag.au/link/ac2k>
<https://pemag.au/link/ac1n>

Other stuff

Soldering guide
 Basic multimeter

<https://pemag.au/link/ac2d>
<https://pemag.au/link/ac2f>

Components for Weird & Wonderful Projects, part 1

4-inch (10cm) jumper wires (optional)
 8-segment DIP red LED bar graph displays

<https://pemag.au/link/ac2l>
<https://pemag.au/link/ac2c>

Components for Weird & Wonderful Projects, part 2

144 tricolour LED strip (required)
 Pushbuttons (assorted colours) (required)
 7-Segment display modules (recommended)
 2.1mm panel-mount barrel socket (optional)
 2.1mm inline barrel plug (optional)
 25-way D-sub panel-mount socket (optional)
 25-way D-sub PCB-mount plug* (optional)
 15-way D-sub panel-mount socket (optional)
 15-way D-sub PCB-mount plug^ (optional)

<https://pemag.au/link/ac2s>
<https://pemag.au/link/ac2t>
<https://pemag.au/link/ac2u>
<https://pemag.au/link/ac2v>
<https://pemag.au/link/ac2w>
<https://pemag.au/link/ac2x>
<https://pemag.au/link/ac2y>
<https://pemag.au/link/ac2z>
<https://pemag.au/link/ac30>

* one required for each game cartridge you build

^ one required for each auxiliary control panel you build

Components for Weird & Wonderful Projects, part 3

22 AWG multicore wire kit
 20 AWG multicore wire kit

<https://pemag.au/link/ac3m>
<https://pemag.au/link/ac3n>

Components for Weird & Wonderful Projects, part 4

5V Buck converter module
 9V 2A power supply (optional)
 Bench power supply (optional)

<https://pemag.au/link/ac2m>
<https://pemag.au/link/ac46>
<https://pemag.au/link/ac48>

to do is enhance our switch debounce solution to address all the problems I haven't told you about yet.

Until then, if you have any thoughts

you'd care to share on anything you've read here, please feel free to drop me an email at max@clivemaxfield.com.

And, as always, have a good one! **PE**

PoLabs

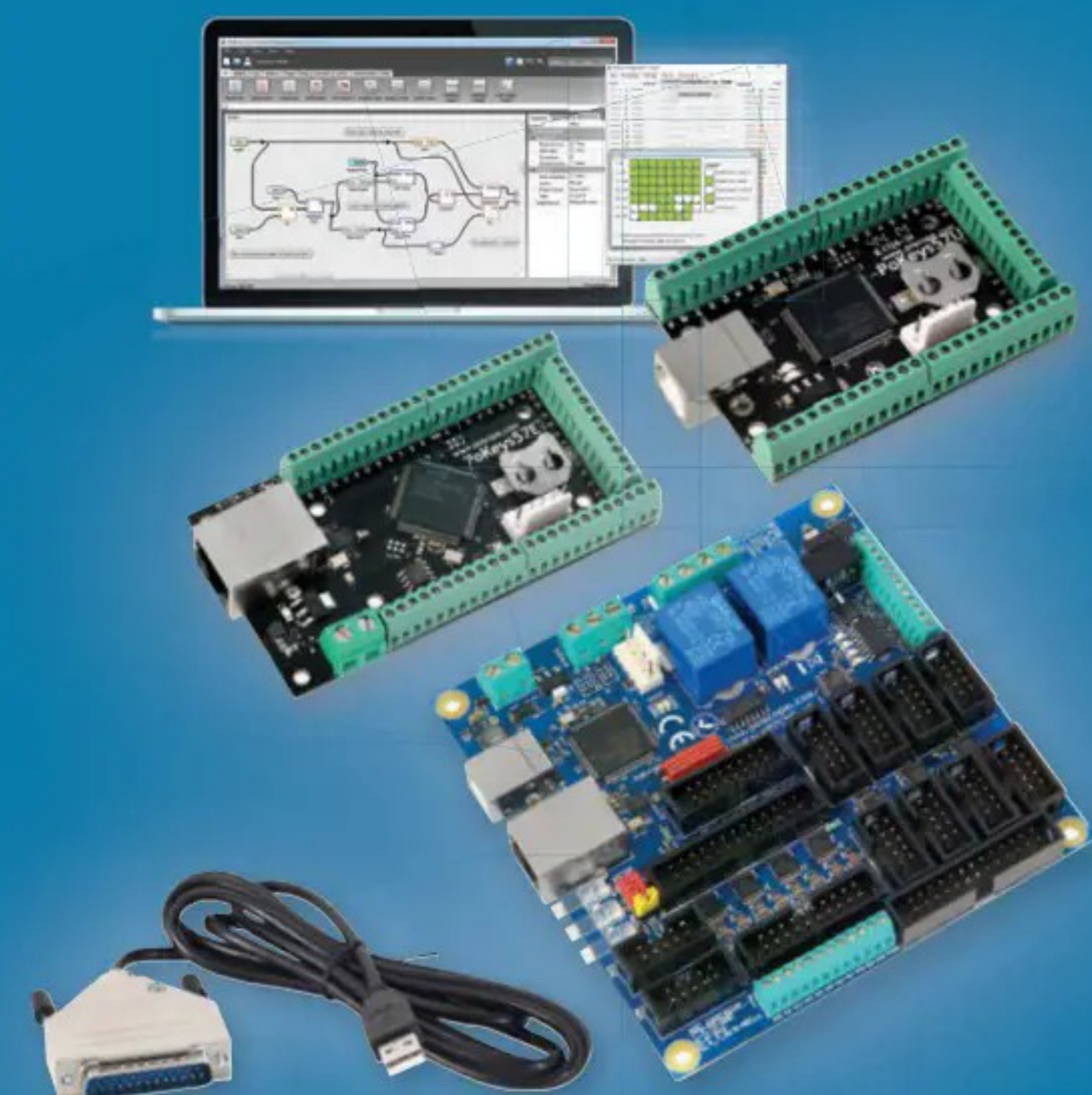
For more information or software download please visit



www.poscope.com/epe

PoKeys

Connect, Control



- USB
- Ethernet
- Web server
- Modbus
- CNC (Mach3/4)
- IO
- PWM
- Encoders
- LCD
- Analog inputs
- Compact PLC

Stepper motor drivers

Drive



- up to 256 microsteps
- 50 V / 6 A
- USB configuration
- Isolated
- up to 32 microsteps
- 30 V / 2.5 A

PoScope Mega1+ PoScope Mega50

Measure



- up to 50MS/s
- resolution up to 12bit
- Lowest power consumption
- Smallest and lightest
- 7 in 1: Oscilloscope, FFT, X/Y, Recorder, Logic Analyzer, Protocol decoder, Signal generator

Constructional Project

Background image: <https://unsplash.com/photos/gaming-room-with-arcade-machines-m3hn2Kn5Bns>



This retro game is a fun and educational project to create a dexterity-based game with nine difficulty levels, a health bar graph, a timer and four different sound effects. It is based mainly on 4000-series logic and all through-hole parts on a single circuit board.

There are some sections of this game that advanced constructors can customise, such as changing the winning and losing tunes, but we will describe the assembly process for the standard version. There is definitely scope for customisation when you make the wire 'obstacle course', as you can make it as easy or as hard as you want!

The assembly instructions will be given in seven discrete steps. After adding the parts for each step, you will have new functions to test, so you can pick up any problems early on and fix them before tracking them down will be more difficult.

You shouldn't need any special tools; a soldering iron, solder, fume extractor and a multimeter for testing should be all that's required.

Construction

We will build the Skill Tester 9000 section by section and test each as we go along. This allows people to work with young constructors or students in simple sessions, achieving visible progress in each. Even as an experienced constructor, I build projects in

bite-sized chunks as it makes debugging simpler and there are built-in coffee breaks.

The Skill Tester is built on a double-sided PCB coded 08101241 that measures 174 × 177mm. During construction, refer to the PCB overlay diagram (Fig.5) and Photo 1 to see which components go where and how they are orientated.

Here are general cautions and instructions you should keep in mind during the construction process:

- Ensure all diodes are fitted the right way around (stripe to the right or down on this PCB).
- All LED cathodes are upwards; LEDs have a chamfer (flat edge) on the cathode side.
- All ICs are installed with pin 1 to the right. If you get one backwards and are not using sockets, you will have to cut all pins off using side cutters and pull individual pins out (unless you have a hot air rework station).
- Check the supply rail voltage every time you power it up after adding parts. We have included a ground test point, with a 9V test point close by (below

IC1). You should measure more than 8V between the two with a fresh battery. If not, something is wrong.

- If something is wrong and, while you are investigating, the noise from the speaker is slowly driving you insane, put a 1kΩ resistor in series with the speaker to tone things down a bit.

- Standard checks as you solder: are there any solder blobs shorting pins? Is each solder joint shiny with the right shape? Has the solder adhered to both the component lead and the PCB pad?

- For each polarised part, check before and after soldering that it is the right way around. Also check the part numbers of ICs and double-check the orientation before you solder them.

- If you need to check clock signals and don't have an oscilloscope, put your DVM on its AC setting and probe the test point. You should measure a few volts AC or see pulses in the reading for very slow clocks.

Touch, Win & Reset sections

1. Let's start construction with the

Skill Tester 9000, part two

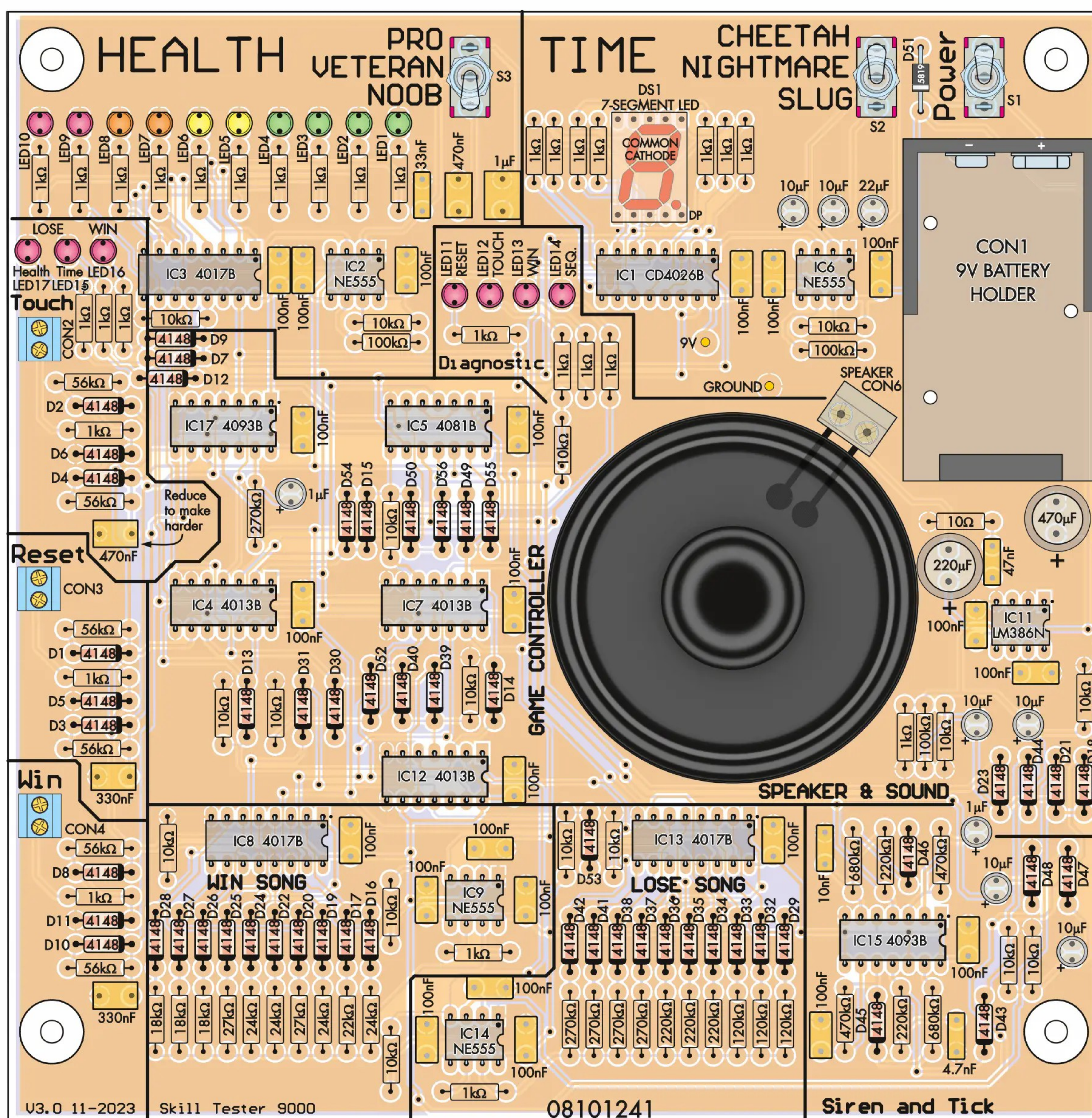


Fig.5: this overlay diagram shows which components go where. It also shows the correct orientations of all polarised components like ICs, diodes and electrolytic capacitors. It is divided into 11 sections and can be assembled all in one go if you are confident and experienced, or in the seven steps outlined in the article text.

Touch, Win and Reset circuits. Fit all the parts in the Win, Reset and Touch areas of the PCB, plus LEDs LED11-LED14 and the four associated 1kΩ series resistors.

Also install IC17 (4093B), the 100nF capacitor next to it, power switch S1, 1N5819 diode D51, the battery holder and the 470μF capacitor just below the holder. When inserting the DIP ICs into the board, you may have to bend their leads inward a bit as they will come splayed outwards.

Put a battery in the holder and check that the 9V rail is OK. If not, is D51 the right way around? Is S1 switched on?

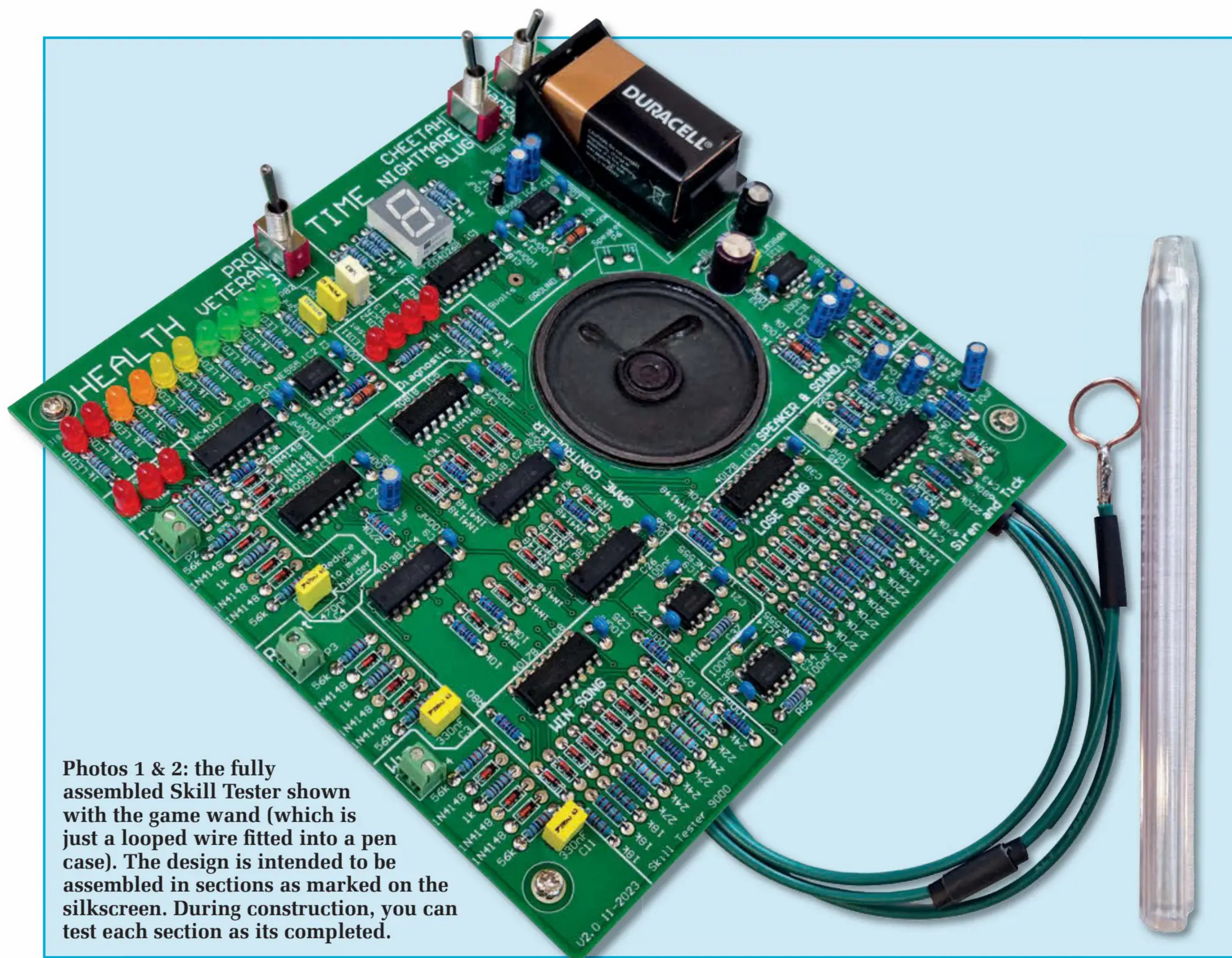
Now short the terminals of CON3 (eg, using a short length of wire). You should see the Reset LED (LED11) turn on. Repeat this for the Win and Touch inputs at CON4 (with LED13 lighting) and CON2 (with LED12 lighting).

If that does not work, are IC17 and the diodes the right way around? Measure the voltages on the input

connectors; one terminal should be at 0V, while the other should be pulled up to 9V.

If only one input does not work, look for solder bridges, especially on the pins of the 4093 chip, IC17. Check that its pin 2 goes low when you short the Reset pads and pin 3 goes high. Verify that its pin 9 goes low when you short the Touch pads and that pin 10 goes high. Also test that its pin 6 goes low when you short the Win pads and pin 4 goes high.

Constructional Project



Photos 1 & 2: the fully assembled Skill Tester shown with the game wand (which is just a looped wire fitted into a pen case). The design is intended to be assembled in sections as marked on the silkscreen. During construction, you can test each section as its completed.

At this point, you should be able to trigger Reset and exercise the Touch and Win inputs and see the corresponding debug LEDs light.

Health section

2. Now let's build the Health Clock and LEDs. Fit all the parts in the PCB section labelled HEALTH. Do not miss the 10k Ω resistor just below IC3.

Get the selection of your coloured LEDs for LED1-LED10 right! We used low-cost LEDs with similar brightness. Make sure the switch you use for S3 is a centre-off type, so you have three difficulty levels.

Apply power to the board and check that the 9V rail is OK. You should see LED17 come on. Short the pads of the Touch input (CON2), and you should see LED1-LED10 cycle continuously. Short the RESET pads (CON1), and LED17 should relight.

If the above works, great, let's move on. If it does not work, check that there is a clock signal at pin 14 of IC3. If the

clock signal is missing, check around IC2. Short the Touch pads and check that pin 4 of IC2 goes high. If it still isn't working, Check that IC3's CP0 (clock enable) pin (pin 14) is low. If not, is there a short, or did you forget to fit the 10k Ω resistor? Are your LEDs the right way around?

Check the Out Of Health signal on pin 12 of IC3 (4017); it should produce a square wave once per LED cycle.

Check that the debug LEDs for Touch and Reset still work; if not, retest the input circuitry. At this point, LED1-LED10 should be cycling continuously.

Time section

3. Next comes the Time section and its 7-segment LED display. Fit all the parts in that area. Ensure that all parts in the Time and Health areas are on the board now. Verify that S2 is a centre-off type so we get three difficulty levels.

Apply power to the board and check that the 9V supply is OK. The

7-segment display should count from 0 to 9 continuously. If there are any problems, use a similar testing procedure to the section above, but with IC6 and IC1.

The 555 (IC6) clock output is at pin 3, and the 4026 (IC1) clock input is at pin 1. The clock inhibit pin on the 4026 (pin 2) should be low, and Reset pin 15 on the 4026 (IC1) should be low. Shorting the reset pads (CON3) should reset the counter to 0. The pin 5 carry output of IC1 should cycle high and low once per 0-9 count. The Time counter should be running continuously unless you trigger Reset.

Siren and Tick section

4. Fit all the parts in the Siren and Tick area of the PCB. Watch out, as the 1 μ F capacitor may look the same as the 10 μ F capacitors. Also solder IC5, the 4081B in the Game Controller section and the parts immediately around it: the 100nF capacitor next to it, two 10k Ω resistors (one to the right

Skill Tester 9000, part two

and one below IC5) and the six diodes immediately below IC5.

Mount the speaker by gluing it in place with a few small dabs of super glue, silicone sealant or Araldite. Keep it tidy (ie, avoid getting glue where it shouldn't go). Wire the speaker to the pads for CON6; you can omit the actual screw terminal or mount it on the underside of the board.

Apply power and check the 9V rail. You should hear a warbling 'ping' from the speaker; that is the Time Clock tick. Switch the Time switch between Slug, Cheetah and Nightmare. You should hear the ticks change pace from very slow to very fast.

If there is no sound, probe pins 10 and 11 of IC15 with an oscilloscope or multimeter reading AC volts. There should be AC signals on both.

If so, short out diode D43, and you should get a lot of noise from the speaker. In that case, there is something wrong with C44, D47, C46 and the associated parts. Are those capacitors the right way around?

Now short out the Touch pads (CON2). You ought to hear a racket from the speaker (the Touch tone). If not, check for an AC signal on pins 3 and 4 of IC15.

The signal at pin 3 will have a low frequency, so you will be able to detect the individual pulses. Check that the Touch LED (LED12) lights when you short the Touch pads. If it still isn't working, check for shorts on the board and parts missing or the wrong way around.

At this point, the ticking sound should be running non-stop, and the Touch tone should be generated if you short the Touch input pads (CON2).

Win Song section

5. Now fit all the parts in the Win Song section. There are a lot of different value resistors in the tune section; double-check the value of each before you solder it in. Getting resistors off a double-sided board is possible but not easy.

If you are not 100% sure, measure each resistance with your multimeter. Remember to avoid touching the meter probes when doing this, as that will affect the measurement of high resistances.

The 1 μ F capacitor and 270k Ω resistor just below IC17 should also be fitted to the PCB now. They set the period of the tune sequencers.

Apply power to the board and check the 9V rail. You should hear the ticking timer noise and the Win Song playing repeatedly. If there is no sound or only a single tone, check IC8 and IC9 for solder bridges. Also check around IC17, as it generates the clock for the tunes.

Probe pin 14 of IC9 with an oscilloscope (or multimeter on AC volts). You should find a signal at about 2Hz.

Check pin 3 of IC9 (555 timer). It should have an audio-frequency AC signal on it. Are those capacitors and diodes the right way around?

At this point, you should have the Win Song running continuously on top of the ticking sound.

Lose Song section

6. Fit all the parts in the Lose Song section, then apply power and check the 9V rail. You should hear the time 'ping' with a crazy noise in the background, which is the Win Song and Lose Song playing on top of each other.

Short pins 1 and 4 of IC9 (555) to stop the Win Song so you can hear the Lose Song by itself. Be careful not to touch any other pins or parts, while doing this.

If that makes no difference, probe pin 14 of IC13 with an oscilloscope or multimeter on AC volts. You should see a signal at about 2Hz.

Pin 3 of IC14 (555) should have an audio-frequency AC signal on it. Are those capacitors and diodes the right way around?

You can stop the Win and Lose tunes independently by connecting a wire to the ground point, then touching the other end to pin 4 of IC5 or IC14, resetting that 555 and stopping that tune generator.

At this point, you should have some crazy noises happening while power is applied.

The Game Controller

7. We have built and tested each part individually, and you should understand how each section operates. Let's bring them all together by adding the control components. Fit all the remaining bits, with the usual cautions on getting ICs in the right way around and choosing the right one for each spot.

Apply power and check that supply voltage again. The game will start straight off the bat. You should hear ticking, and if you short the Touch pads (CON2), you should hear the Touch noise.

You should hear the Lose Song after the Time counter gets to 9. Try shorting the Reset pads (CON3), which should restart the game. If you short the Win pads (CON4) immediately after starting a game, you should hear the Win Song.

If that is not happening, verify that all polarised parts are fitted the right way around, especially the diodes. Check for bad solder joints or bridges (shorts) between adjacent pads, or components that have been mixed up or misplaced.

The diagnostic LEDs (LED11-LED17) show the state of every latch and input. Our earlier tests showed that the inputs were working, so check everything around IC4 and IC7, as the latches are there. Are any pins shorted?

Mounting the PCB

Our baseboard was about 500mm long and just deeper than the PCB. How you go about this part of the construction process should reflect the space you have and what you want this game to be.

We marked the holes for drilling by putting the completed PCB on the board and drawing through the holes with a marker, then drilling 4mm holes at those points. We had shorter screws on hand, so we countersunk the holes on the back of the game board to get a bit of extra length so our screws reached the standoffs on the top side of the board.

Once you have successfully mounted the PCB, remove it to work on the game wire, including the Reset and Win parts at each end.

We routed the edges of our board and painted it with clear lacquer to make it tidy. The charm of this project is its nostalgic design and concept, which relies in no small part on a tidy appearance.

You need to put some rubber feet on the board, in the corners of the underside. Otherwise, the screw heads will scratch everything you put it on, and it will slip around. Stick-on rubber feet work well.

The game wire

We used some fencing wire from the shed for this. It is about 2mm thick and pretty solid; it can be bent with pliers or your hands for smoother curves but is tough enough to take a hiding. You want to use steel wire as

Constructional Project

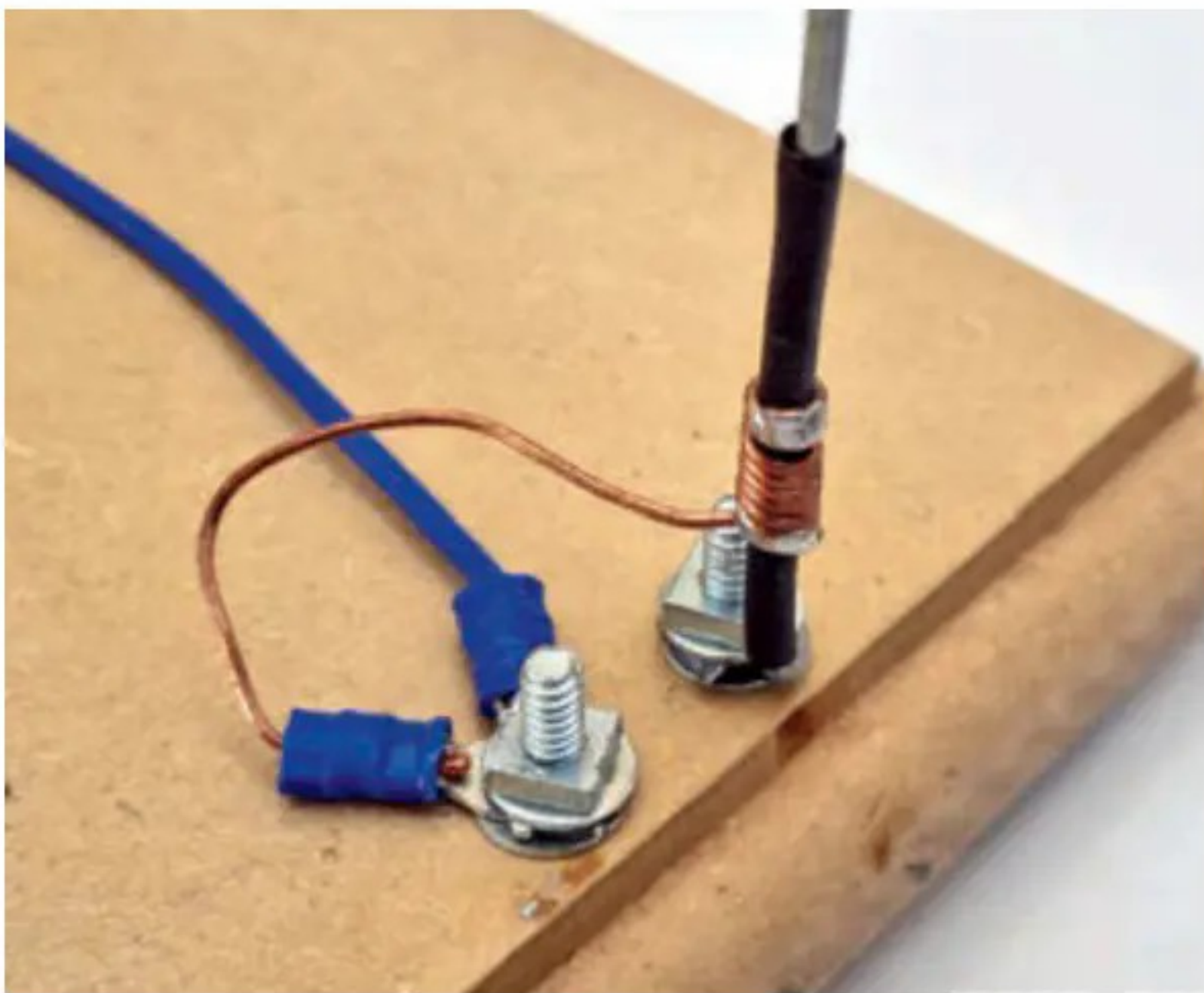


Photo 3 (left): this photo shows how we secured the game wire to the baseboard. It also shows the wire wrapped around it to form the Win contact (with heatshrink tubing underneath) and how that copper wire connects back to the terminal on the PCB. To attach it to the baseboard, the main wire was bent into a loop just larger than the bolt diameter using sharp-nosed pliers.



Photo 4 (right): a view of the Reset end of the maze wire, showing how the start contact is bare copper wire from domestic mains cable, wrapped around the heatshrink insulation and soldered in place. Tinned copper wire would also work here. This is also the point that the main game wire is electrically connected to the Touch terminal on the PCB.

copper won't spring back. It doesn't need to be fancy or new. If you are scratching your head, look in your wardrobe for a metal coathanger. I reckon that would work just fine.

The shape of the maze is up to you. The tighter the kickbacks and the more changes in direction, the harder the game will be. A loop makes the game super hard as the wand has to be rolled to the back of the game to achieve this; that might be for more advanced gamers.

Photo 3 shows how we bent the wire to go through the screws on the game board.

We drilled holes for 30mm “gutter bolts” about 30mm apart, allowing space for a bolt to hold the game wire and a spot for a second bolt to connect to the Win and Reset wires at each end of the maze. We did this to make the whole thing robust, and so we could pull the wire maze off and put in an easier or harder one later.

Our first maze had loops, tight corners, and all sorts of complex curves, making it nearly impossible to play. We suggest you instead start simple and work from there.

Once you have your maze bent up, but before you bend the loops for the bolts, insulate about 50mm at each end with a couple of layers of heatshrink tubing. That will allow us to wrap 10-20mm of bare copper or tinned copper wire around the outside to form ‘pads’ that we will connect to the Reset and Win inputs on the game board.

These inputs have a pullup on the game board, so if we touch these pads with a grounded wire, we will trigger

Reset or Win, respectively. Our wand will connect to ground, making a neat arrangement for these inputs to the game. The required wiring is shown in Fig.6, although it does not show the physical layout, just what connects where.

After you have applied the heatshrink tubing, tightly wrap your copper wire around it. Use pliers to ensure it is tightly in place, then solder the top and bottom of the loops together. Don't worry; the heatshrink will survive; we put two layers just to be sure. This is shown in Photos 3 & 4. We left enough wire to run to a bolt where we connect Reset and Win to flying leads from the game board.

The wand

For the wand, we want something that is comfortable to hold and to which we can fix the wire loop that goes around the maze wire, connected to our circuit ground.

By connecting the loop on the wand to ground, we can tap this on the Reset wire to start the game. If the loop touches the maze wire, it connects to Touch and, at the end of the game, tapping on the Win wire wins the game; both are part of the maze wire.

We used a ballpoint pen case for the wand (Photo 5). The loop was made from the same wire, reclaimed from house mains wiring, that we used to make the Reset and Win pads.

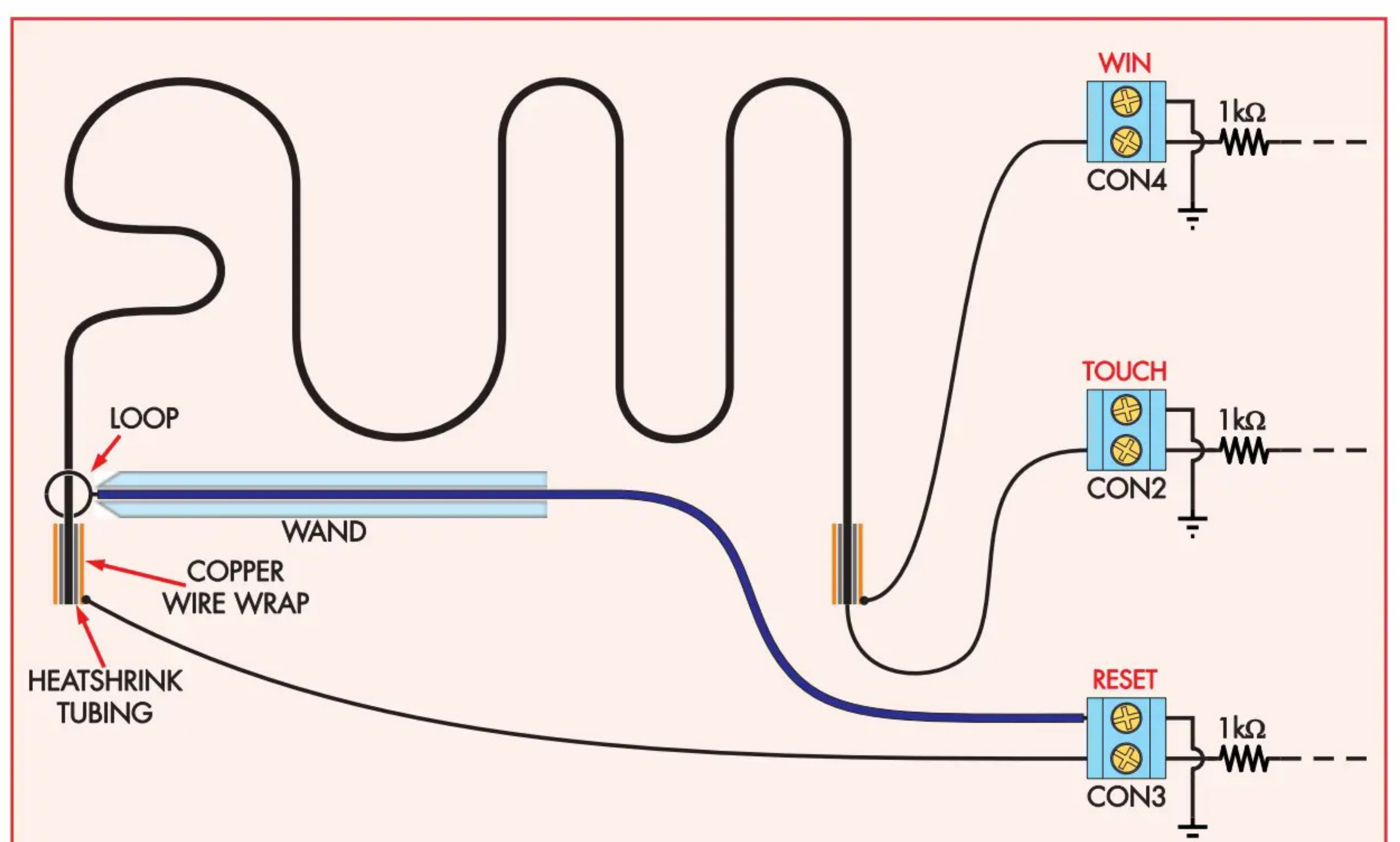


Fig.6: this diagram shows how the Touch, Reset and Win terminals (CON2-CON4) connect to the wand, game wire and start and finish pads. Refer to the photos to see how we made the required electrical connections, and note that the ground wire going to the wand can connect to the upper screw of any of the three terminals.

Skill Tester 9000, part two

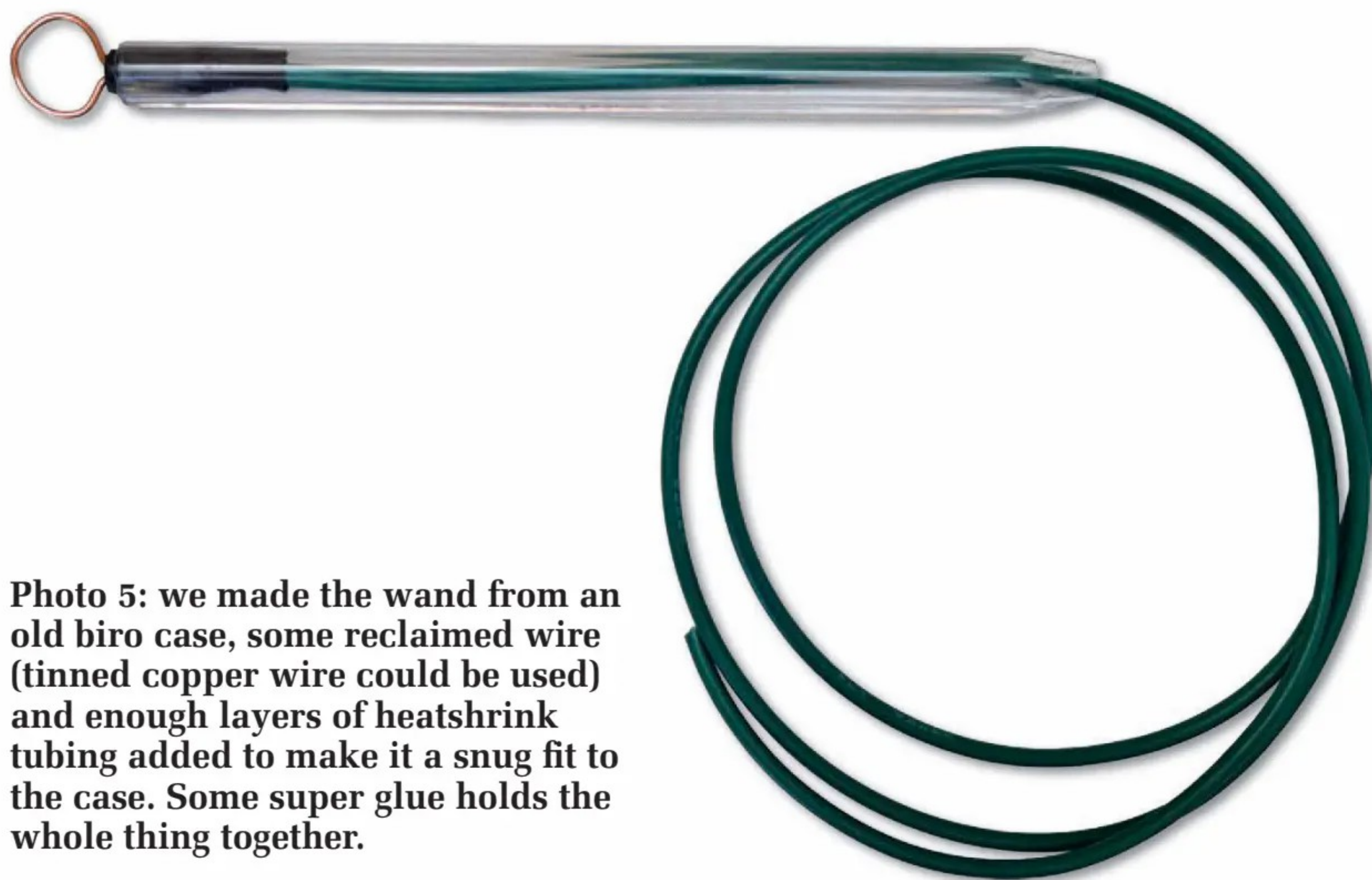


Photo 5: we made the wand from an old biro case, some reclaimed wire (tinned copper wire could be used) and enough layers of heatshrink tubing added to make it a snug fit to the case. Some super glue holds the whole thing together.

Assembly is a simple matter of screwing everything together. Make sure that the wand's loop is ultimately wired to a ground pin on one of the Reset, Touch or Win connectors (CON2, CON3 or CON4). In each case, the ground side is closer to the top of the board.

By now, you will realise there is a bit of work in making this project and doing so tidily. Still, the basics of a stable base, something to screw the wire to and wide enough to hold the PCB are the essence (see Photo 6 for our completed version).

Finishing it off

Connect the pad at the start of the

wire to the Reset line (not the ground side, so the bottom terminal of CON3) so that tapping the wand here will start the game.

Connect the main wire to the Touch connector (bottom of CON2), so touching the wand to the wire will short the Touch pin and reduce the Health counter.

Connect the pad at the end of the wire to the Win connector (bottom of CON4). This way, everything you need to run a game is at your fingertips.

Tips on playing

There are three settings each for speed and difficulty. Noob + Slug makes the game the easiest, while Veteran +

Nightmare makes it the most difficult (perhaps impossible)! So start with Noob + Slug and work your way up from there. To win, you must move the wand from the start to the end of the wire with time and health left.

To play a one-on-one game, choose a difficulty setting and play one game each. If one player wins and one loses, the winner is obvious, but if both win, whoever has the most health left wins. If both have full health left, the fastest time wins.

To run a tournament, start with the easiest settings and give each player one attempt at the game. Anyone who loses (whether by running out of time or health) is eliminated.

If more than one person is left, play again on a higher difficulty setting. Repeat this until all but one has been eliminated, or you reach the highest difficulty setting. In the latter case, use the rules above to determine the winner.

When increasing the difficulty, we suggest going from Noob to Veteran for Health first, then when you reach Veteran, start speeding up the time from Slug to Nightmare.

Also, remember that the way you bend, fold and make loops and kick-backs in that wire plays a big part. Is your wire tough enough?

Have fun! If you come up with better tunes than we have, send in your resistor values so we can try them ourselves! **PE**

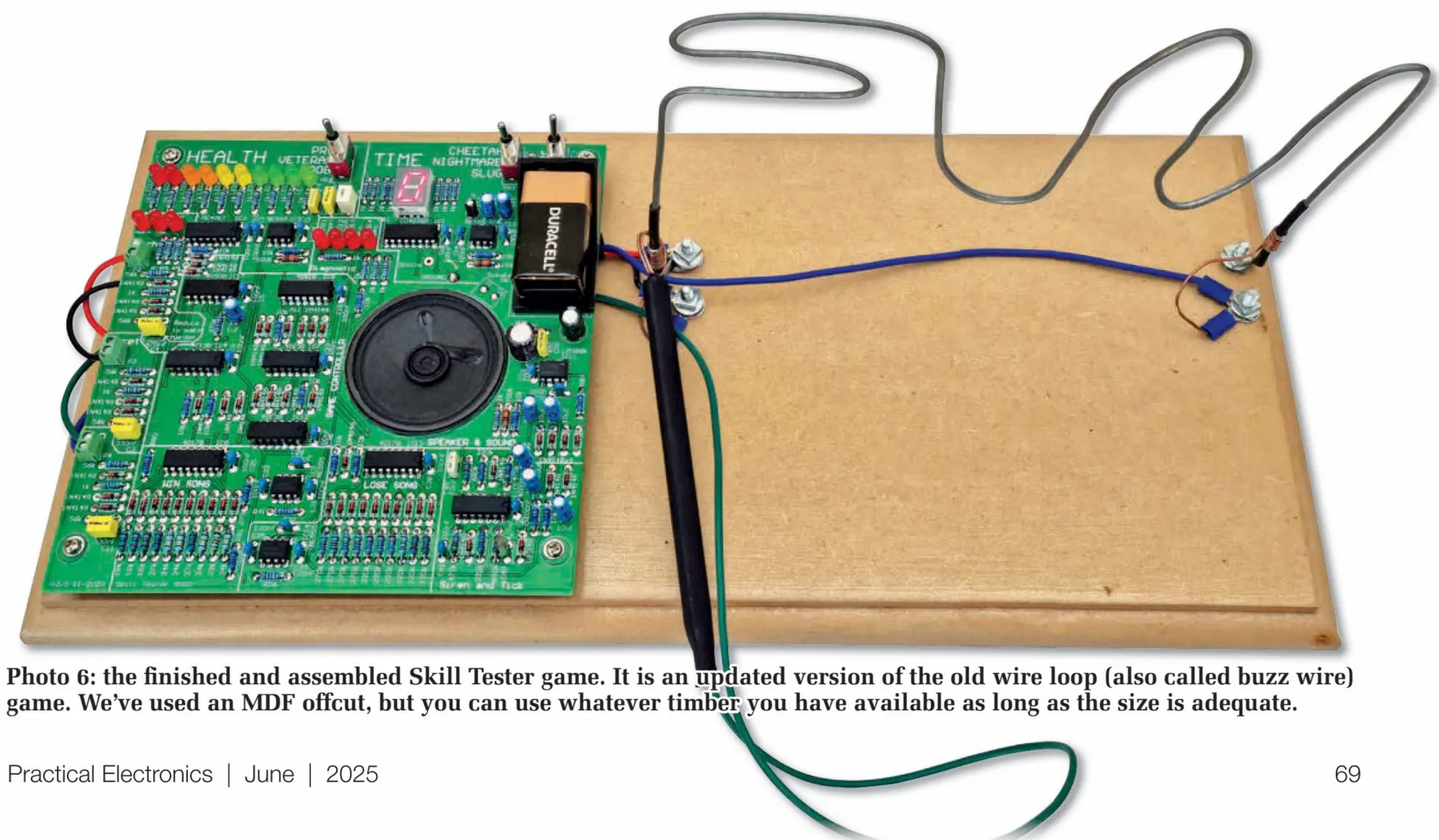


Photo 6: the finished and assembled Skill Tester game. It is an updated version of the old wire loop (also called buzz wire) game. We've used an MDF offcut, but you can use whatever timber you have available as long as the size is adequate.

By Andrew Levido

Precision Electronics

Part 6: Digital-to-Analog Conversion



The first five parts of this series have concentrated on sources of error & imprecision in analog circuitry, such as gain errors, offset voltages, bias currents & noise. These days, most analog circuitry eventually interfaces with the digital realm, such as with a microcontroller, so we also need to understand the precision approach to ADCs & DACs.

In this series so far, we have covered precision electronics from a purely analog perspective. We designed a simple multi-range current-sensing circuit that could be used in a bench-top power supply. It had an analog precision of around $\pm 0.06\%$ at 25°C after trimming out fixed errors.

The idea was that the analog measurement voltage would be digitised via an analog-to-digital converter (ADC), with the trimming and calibration carried out digitally. We will therefore ultimately need to design an ADC subsystem that will have similar or better performance to the analog circuitry we have already developed.

However, the topic of interfacing digital and analog electronics is a big one and quite complex. So I am going to cover it over two articles, starting here with some general principles and then taking a look at digital-to-analog converters (DACs) with a real-world example, before moving on to ADCs next month.

We will also need to take a look at voltage references in a later article, since any discussion of ADCs/DACs would be otherwise incomplete.

Quantisation errors

A digital system like a microcontroller has to represent an analog quantity using a binary-coded number. This can be a simple unsigned integer (zero or a positive whole number) if we are representing a quantity that is always positive, or an offset binary or signed (two's complement) integer if the quantity can be both positive and negative.

Table 1 shows how typical ADCs and DACs represent such quantities, using 8-bit numbers as an example. Notice that the resolution of the two's complement and offset integers are lower because the input span is twice the full scale value.

It might seem obvious, but you need to take this into account when choosing a converter. It's all too easy to fall into the trap of thinking about

full-scale instead of span in these circumstances.

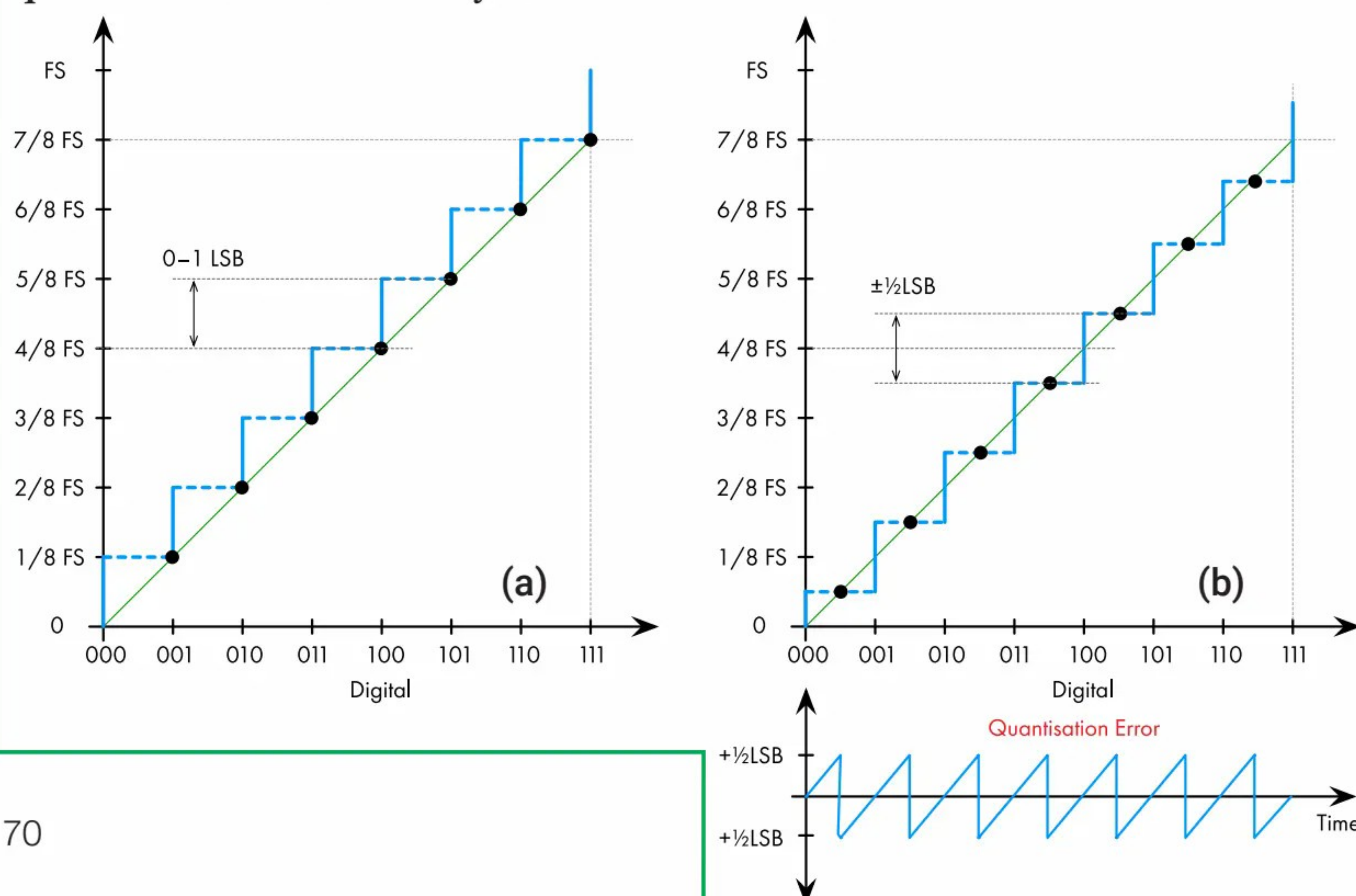
In an ideal converter, the relationship between the analog input or output and the digital code will be as shown in Fig.1. The horizontal axis is the digital code (in this case, three bits for simplicity), while the vertical scale is the analog value as a fraction of full scale.

The green line represents an ideal transfer function, but since the digital codes are discrete, there must be transition points, shown by the black dots.

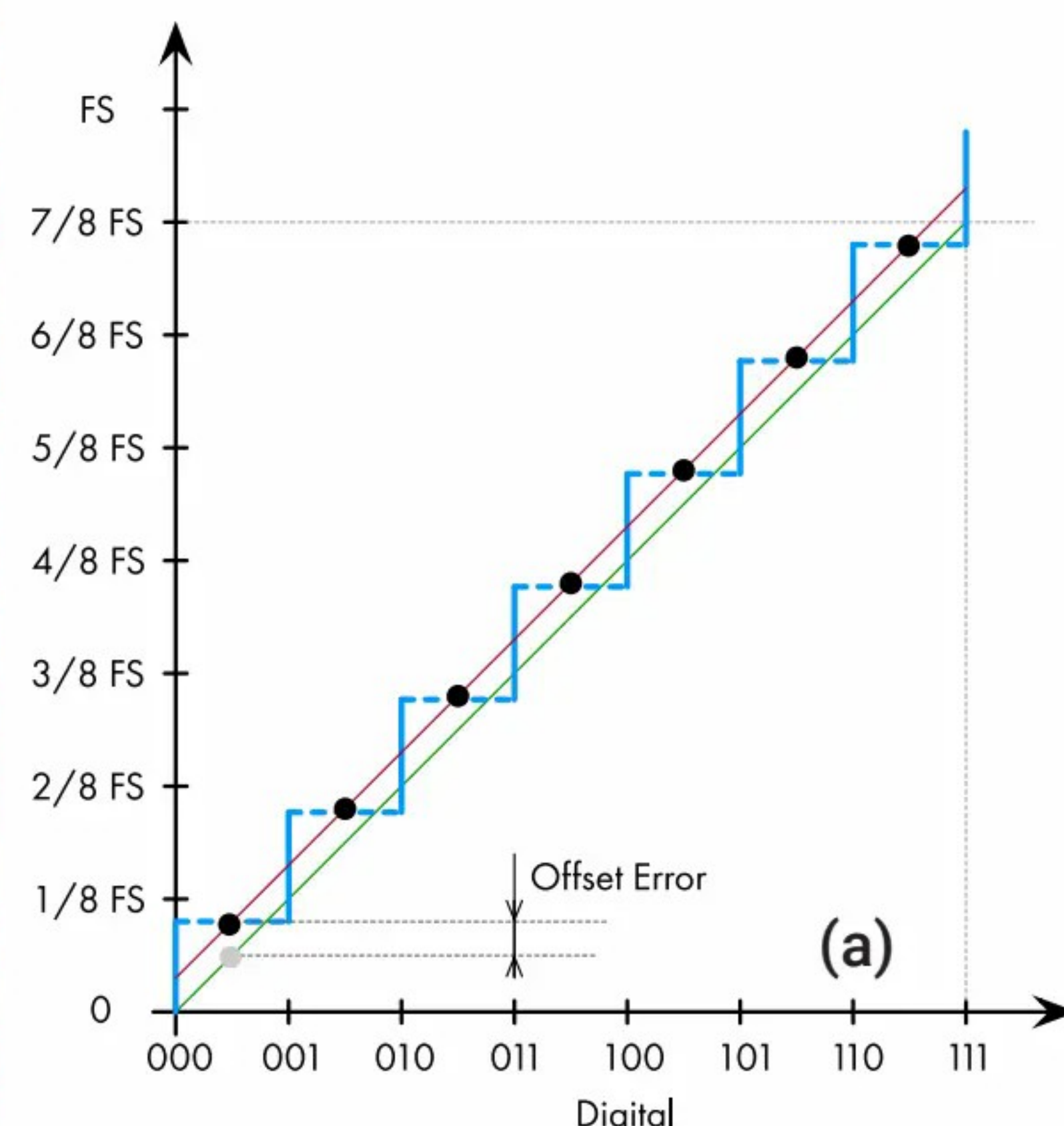
Taking the chart on the left as an example, for an ADC, the code would be zero for an input of zero volts and it would remain so for input voltages up to $\frac{1}{8}$ of full scale, at which point there will be a transition to the 001 code. For a DAC, the output voltage will be somewhere below $\frac{1}{8}$ of full scale if the code is 000.

It's obvious that there is a degree of error inherent in any such converter, since the transition points are

Fig.1: the relationship between analog input or output and digital code for an ideal 3-bit converter. (b) is more representative of real-life devices where the quantisation error is usually $\pm \frac{1}{2}\text{LSB}$.



The vertical axes in Figs.2a-2d are the analog voltages.



Digital-to-Analog Conversion (DACs)

Table 1 – ADC and DAC numeral coding schemes

Voltage	Unsigned Integer	Two's Complement	Offset Binary
+Full Scale	11111111	01111111	11111111
+Full Scale - 1	11111110	01111110	11111110
...	...	...	...
+1	00000001	00000001	10000001
0	00000000	00000000	10000000
-1	-	11111111	01111111
...	...	...	...
-Full Scale + 1	-	10000001	00000001
-Full Scale	-	10000000	00000000

separated by one least significant bit (LSB) of the input code. This is called quantisation error and is an inescapable consequence of the discrete nature of digital systems. In this example, the quantisation error at any point on the transfer function will be somewhere between zero and one LSB.

The chart on the right side of Fig.1 is a more realistic example of how a converter is configured; the ideal transfer function is the same, but the transitions are shifted $\frac{1}{2}$ of one LSB so that they occur between the nominal analog values. In this case, the quantisation error is the same magnitude, but is $\pm\frac{1}{2}$ LSB either side of the nominal value.

Quantisation error can add noise to an AC signal. If we were to apply a linear ramp signal to an ADC, or try to generate a linear ramp with a DAC, the error would look like that shown at the bottom of Fig.1b. The quantisation error would appear as a sawtooth wave with an amplitude of $\pm\frac{1}{2}$ LSB.

We can calculate the signal-to-noise

ratio (SNR) of a digitised sawtooth or triangular waveform. If we have a converter with n bits, the maximum amplitude of an AC waveform can be $\frac{1}{2} \times 2^n$ LSB. The noise amplitude is $\frac{1}{2}$ LSB, so the SNR is $20\log_{10}(2^n)$ or approximately $6.02n$ decibels.

But this only applies to sawtooth and triangle waveforms where the signal has a uniform error distribution. For sinewaves, we have to use the approximation $SNR = 1.76 + 6.02n$ decibels to allow for the uneven error distribution. For an 8-bit converter, the SNR due only to quantising will be around 50dB, and for a 12-bit converter, it will be 74dB.

This is pretty significant; hence, high-fidelity audio ADCs and DACs use many bits and careful filtering to maximise the SNR.

Quantisation error is therefore defined by the resolution of the converter. You obviously need to select a converter with sufficient bits to give you the resolution that your application requires. Typically, you need even

more bits to account for some of the other errors that can occur.

Further ADC & DAC errors

You will see ADC and DAC errors expressed in a range of terms, so it can be a bit confusing at first. We have already seen quantisation error expressed in least significant bits (LSB), but you will also see errors expressed as relative errors (percentages or parts per million) and in absolute terms like millivolts.

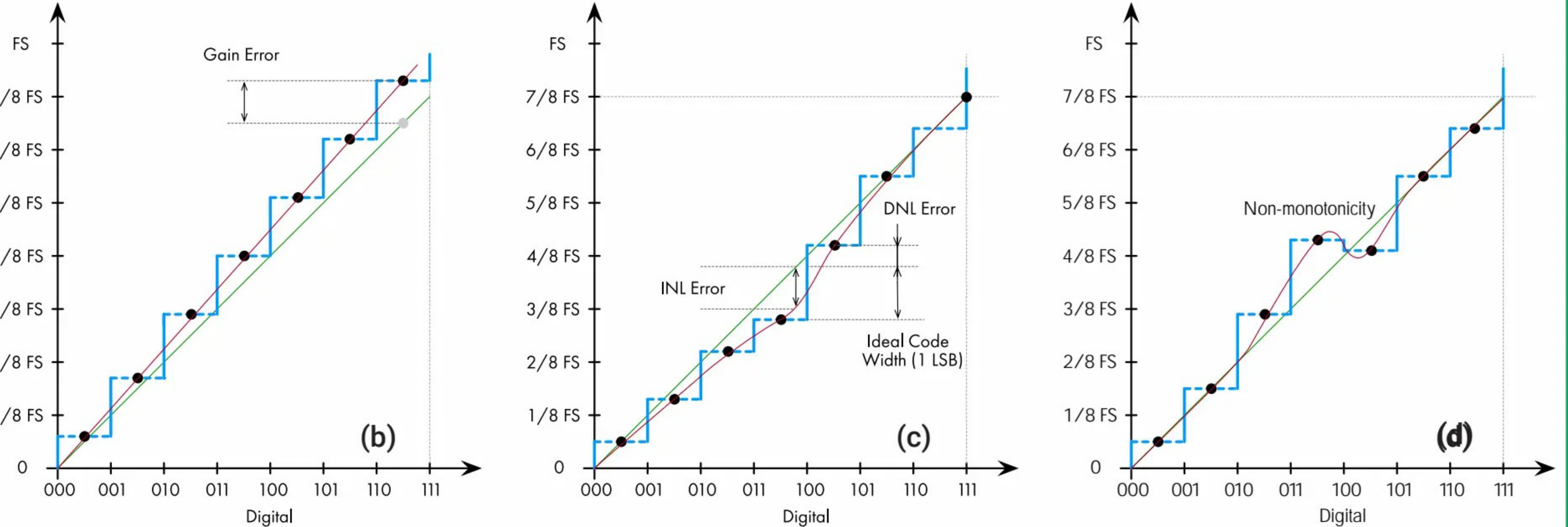
You can convert LSB to a percentage error by dividing it by 2^n (where n is the number of bits, ie, the resolution) and applying the appropriate scaling. For example, the $\frac{1}{2}$ LSB quantisation error on an 8-bit converter will be about 0.2% ($100\% \times \frac{1}{2} \div 2^8$), while on a 12-bit converter, it would be 122ppm ($10^6 \times \frac{1}{2} \div 2^{12}$).

We have seen in past instalments that it can be useful to have errors in both absolute and relative terms, depending on whether we are adding or multiplying uncertain quantities.

Fig.2 shows the four most common types of error that are relevant for ADCs and DACs. Offset error (Fig.2a) is a fixed shift in transition points away from their ideal locations. If measured in LSB, it is defined by the difference between the value of the first code transition and its ideal value. It is most often specified as an absolute voltage, just like an op amp's offset voltage.

As you might expect, there can also be a gain error (Fig.2b) if the slope of the transfer function deviates from the ideal. In LSB, it is defined by the

Fig.2: ADCs and DACs have four main types of error shown in these graphs: offset error, gain error, integral nonlinearity (INL) and differential nonlinearity (DNL). If the DNL exceeds ± 1 LSB, the converter can exhibit non-monotonicity, as shown in (d).



difference between the last code transition point and its ideal counterpart, but it is more often specified as a relative error.

There is also the possibility that the transfer function will deviate from the ideal by not being completely linear (Fig.2c). There are two common measures of linearity error: differential non-linearity (DNL) and integral non-linearity (INL).

INL is the maximum deviation of the transfer function from the ideal over the whole conversion range, while DNL is the maximum difference between the width of a code and its ideal width (1 LSB). A DNL of more than $\pm 1\text{LSB}$ implies a loss of monotonicity, as shown in Fig.2d. A monotonic curve is one that always increases (or decreases).

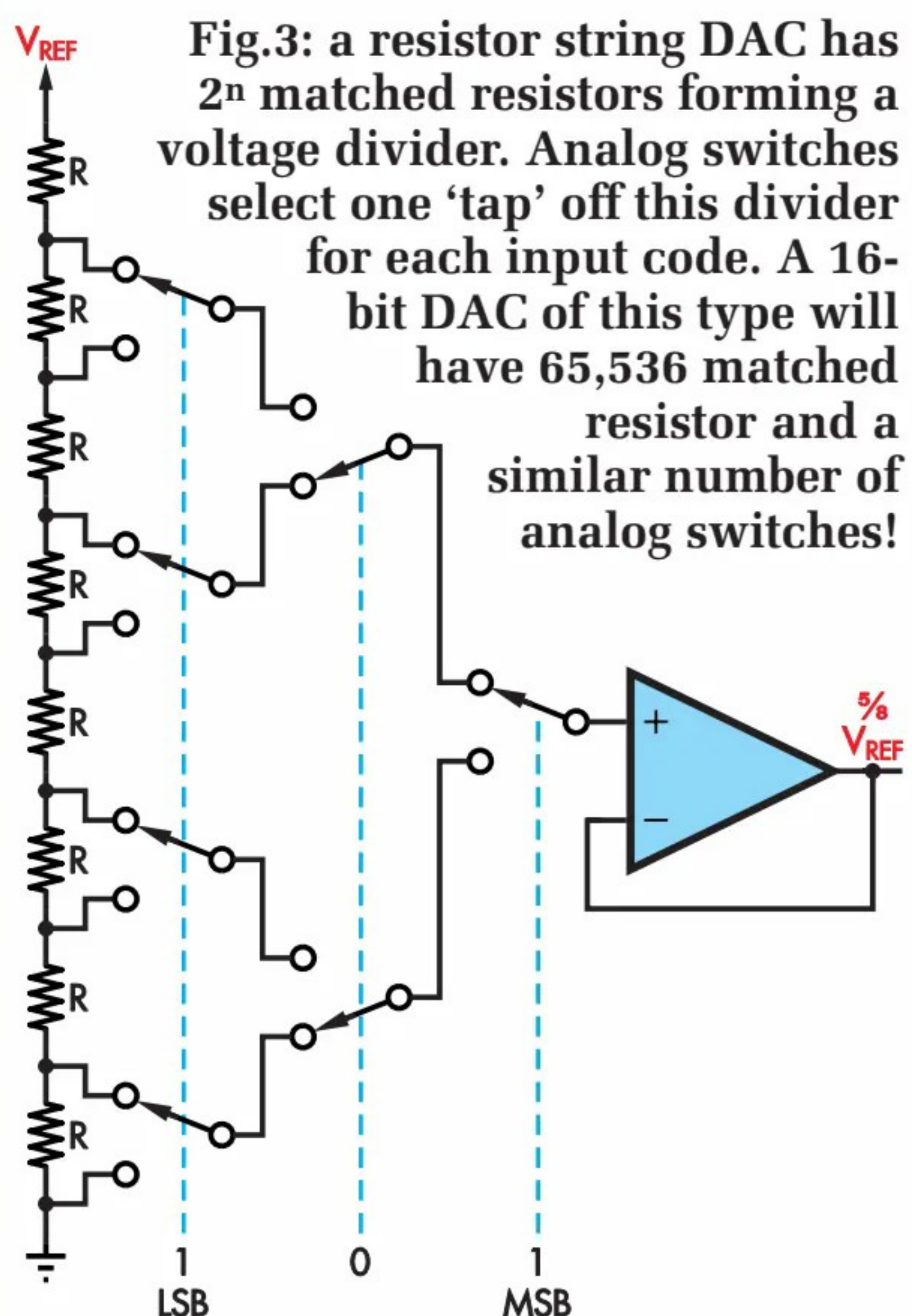


Fig.3: a resistor string DAC has 2^n matched resistors forming a voltage divider. Analog switches select one 'tap' off this divider for each input code. A 16-bit DAC of this type will have 65,536 matched resistor and a similar number of analog switches!

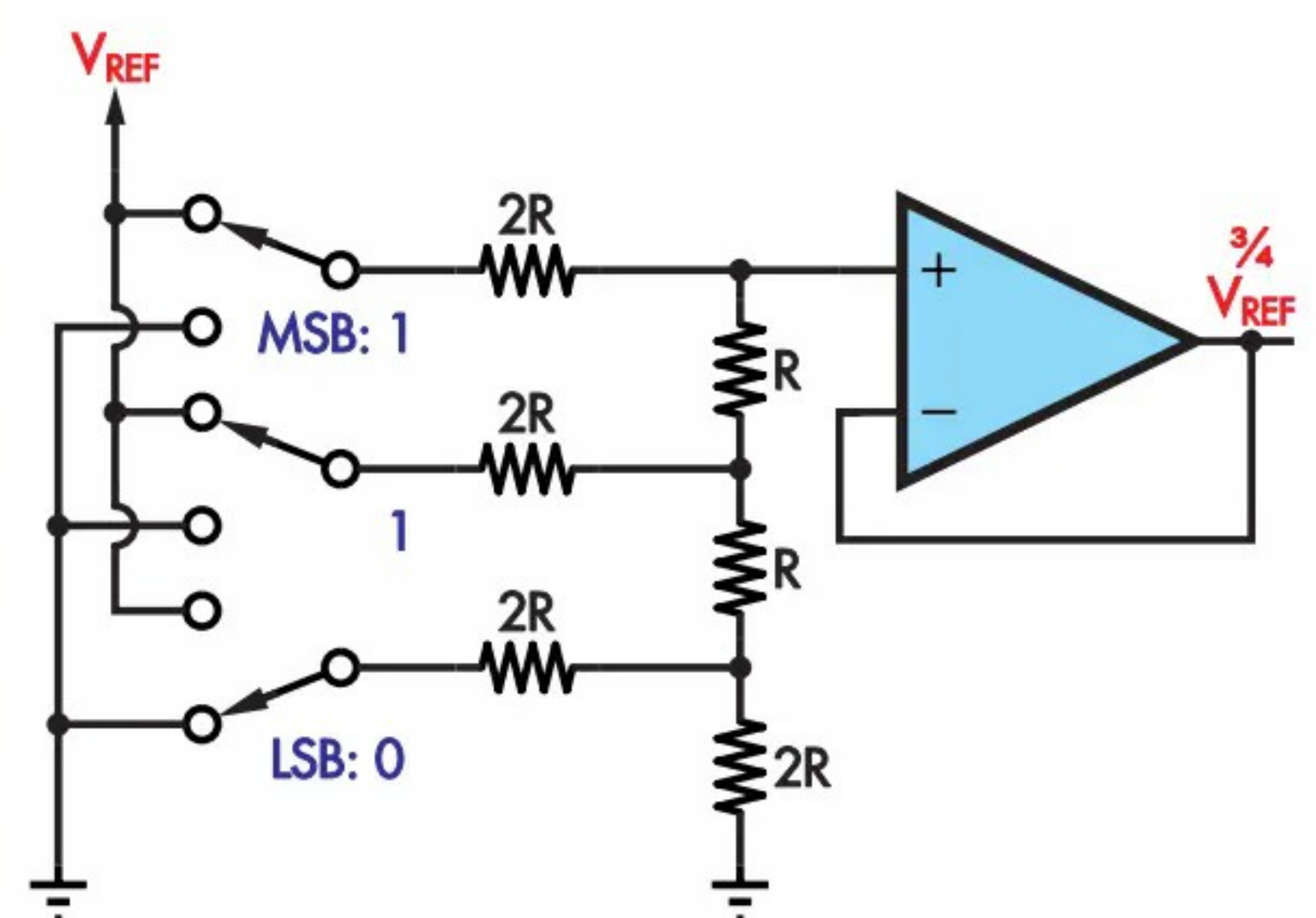


Fig.4: an R-2R DAC uses just $2n$ matched resistors and switches to achieve $2n$ output voltage steps. This architecture is also useful for low-resolution 'roll-your-own' DACs using microcontroller GPIOs in the place of the analog switches.

The INL is the specification to care about if you are looking for the best overall accuracy – for example, to generate or measure a voltage with minimal error. However, if you are using the DAC or ADC in a control loop, you may want to focus on DNL.

The control loop's servo action will look after the INL if it is relatively 'smooth', but 'patches' of inconsistent gain (or worse, non-monotonicity) can cause control glitches like dead spots or even points of instability.

Total unadjusted error

The 'total unadjusted error' (TUE) is a figure that describes the total maximum error for a converter. This is very handy for calculating the error budget. Sometimes manufacturers specify the TUE in the data sheet – either in LSB or as a relative error – but you can calculate it yourself if necessary.

To do so, you convert the offset, gain and INL errors to the same format, and add them using the root-sum-of-squares method (since the error sources are uncorrelated). We will do this for our DAC example later in this article.

Resistor string DACs

Enough theory – let's take a look at a few practical DACs. One common (and fairly obvious) way to construct a DAC is with a resistor string, as shown in Fig.3. A string of 2^n equal-value resistors are used together with a set of binary-weighted analog switches to switch one 'tap' of the string to a buffer and out to an external pin.

The output voltage is $V_{REF}(N \div 2^n)$, where n is the DAC resolution in bits and N is the input code, which ranges from zero to $2^n - 1$.

This type of DAC is guaranteed monotonic by design, and can have quite good linearity since it is possible to match on-chip resistors well. They can also have good temperature stability for the same reason. It is possible to get DACs with up to 16 bits of resolution that use this architecture. The AD5689 we will use in our test circuit is a good example.

This means the chip contains a string of 65,536 matched resistors and a similar number of double-throw analog switches, or equivalent, for each channel. Amazing!

R-2R ladder DACs

The R-2R ladder is a variant on the

resistor string DAC that uses a lot fewer resistors. Instead of requiring 2^n resistors, we can get away with just $2n$, using the circuit shown in Fig.4. Only one double-throw analog switch is required for each bit.

The simplified circuit means you can get R-2R DACs with up to 20 bits or more of resolution. They also can have quite good linearity and temperature coefficients. The output voltage is $V_{REF}(N \div 2^n)$, just like the resistor string DAC.

Another useful property of the R-2R ladder DAC is that you can easily improvise one with an op amp, a handful of resistors and a few digital outputs. The analog switches in Fig.4 effectively switch between V_{REF} and 0V, so they could be replaced with totem-pole digital outputs (say, microcontroller GPIOs), creating a basic 3-bit DAC.

The performance will be average, since the reference voltage will be the digital supply voltage and you will probably use 1% resistors, but if you only need a few levels, this can be a handy technique to create a 'free' DAC.

Current output & multiplying DACs

A variation of the R-2R ladder DAC that provides an output current rather than a voltage allows us to build 'multiplying DACs'.

Strictly speaking, all DACs effectively multiply the reference voltage by the digital code, but many have internal references or external ones that only accept voltages of one polarity. A multiplying DAC can operate in two or four quadrants, as shown in Fig.5.

In both cases, the DACs, shown inside the dashed box, are identical. Since the I_{OUT} pin is sitting at 0V courtesy of op amp IC1, the current coming out of the pin is $N \div 2^n \times V_{IN} \div R$. You can see from the circuit that this equation will hold even if V_{IN} is negative (in which case, the current will flow into the pin).

This current is converted to a voltage by IC1, which is configured as an inverting amplifier. The output voltage of the two-quadrant multiplying DAC will be $-N \div 2^n \times V_{IN}$, where V_{IN} can be positive or negative.

You will notice that the feedback resistor is provided within the DAC IC and is matched to the resistors in

Digital-to-Analog Conversion (DACs)

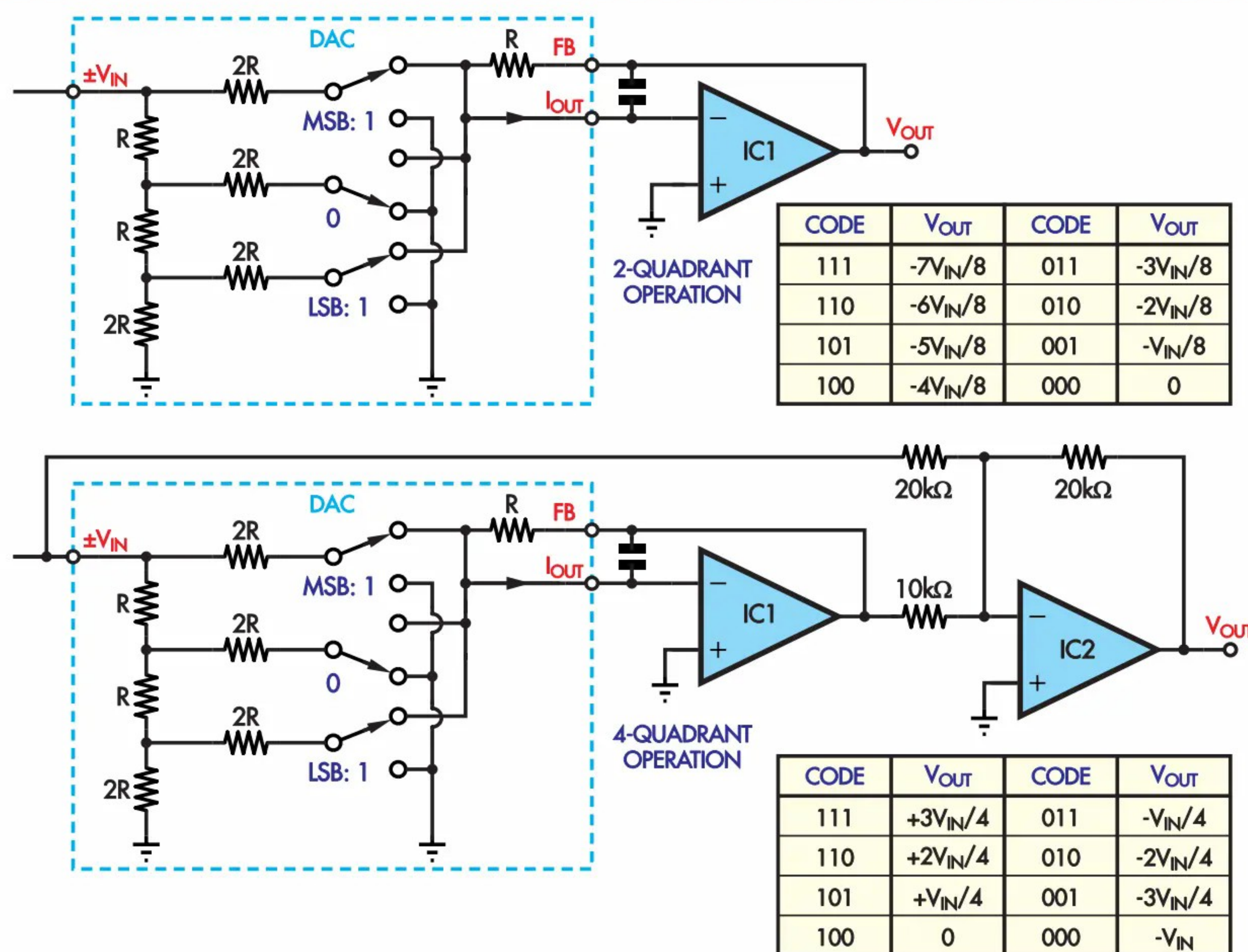


Fig.5: multiplying DACs use a current source architecture to achieve two-quadrant or four-quadrant operation. In both circuits, the reference can be of either polarity or an AC voltage.

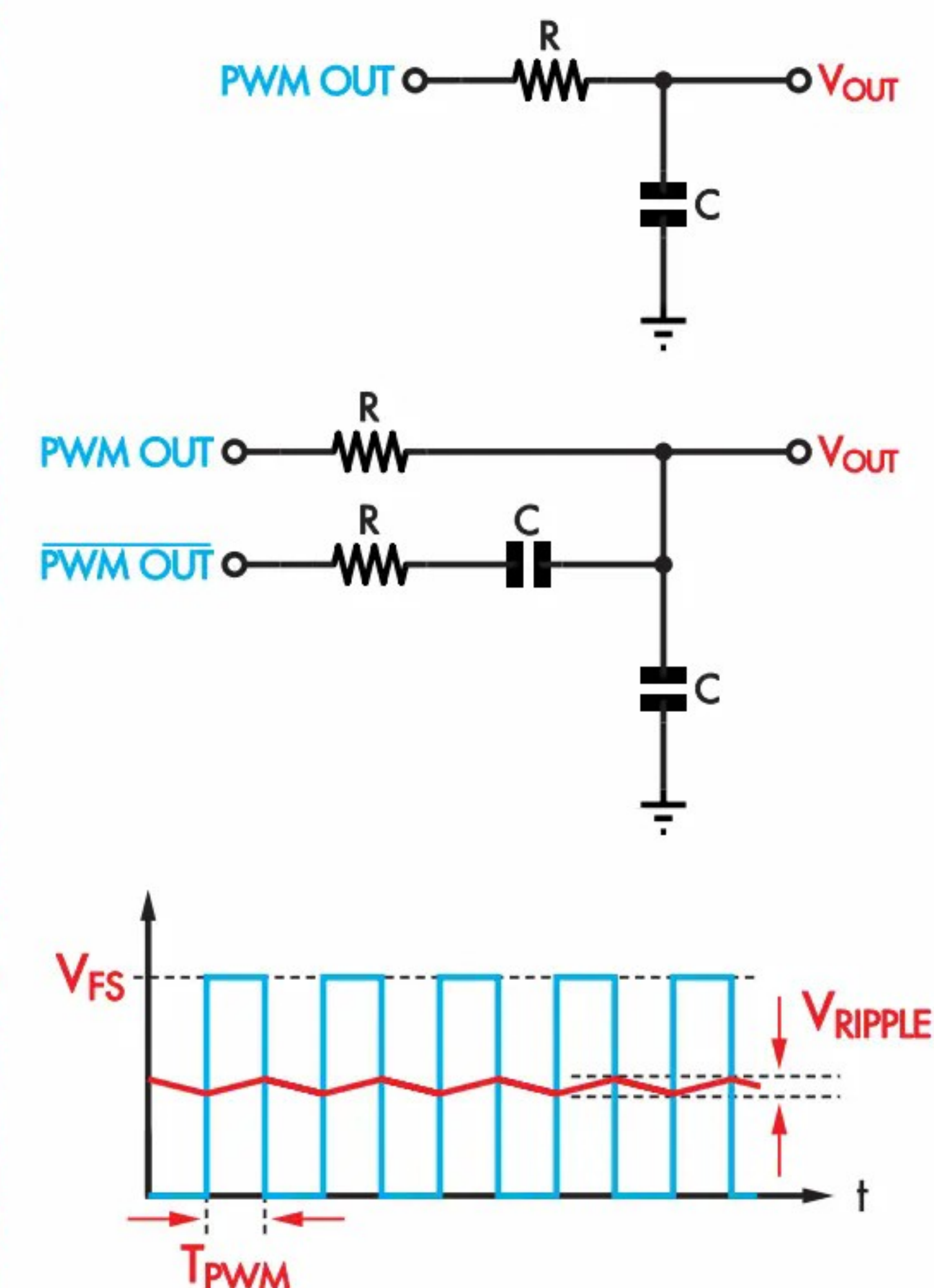


Fig.6: a PWM DAC can be as simple as a microcontroller's PWM output and an RC filter. The lower circuit uses a complimentary PWM signal to improve ripple and settling time eightfold.

the R-2R ladder. This is important because the semiconductor manufacturing process can produce on-chip resistors that are very well matched in value or ratio, but their absolute value is more difficult to control.

Using an external resistor would almost certainly introduce large gain errors and poor temperature stability.

Fig.5 also shows a four-quadrant version of the same circuit, which is identical in operation to the two-quadrant one but has an added (inverting) summing amplifier stage (IC2) that scales up the DAC output and offsets it by V_{IN} . If we consider the code to be a signed value using the offset binary representation, we can effectively multiply a bipolar input voltage by a positive or negative integer.

Multiplying DACs can be very useful in signal processing, for example, as a very fine-grained programmable gain stage. They are also useful in making precise ratiometric measurements.

For example, if you excite some sensor (such as strain gauge) using a voltage produced by a multiplying DAC and digitise the resulting signal with an ADC that uses the same voltage reference, any error or drift in the reference is cancelled. The resulting readings are the true ratio of input to

output independent of the excitation voltage.

Delta-sigma DACs

Another class of DACs worth mentioning are the delta-sigma converters that you frequently encounter in audio applications. Delta-sigma DACs typically have very high resolution (20+ bits) to minimise quantisation noise and have spectacular linearity to ensure low harmonic distortion.

We don't normally use delta-sigma DACs in precision applications because their DC performance is generally not great, probably because this does not matter in audio applications. Oddly, there are plenty of very high-precision delta-sigma ADCs that work very well at DC, as we shall see next time.

Pulse-width modulation

We should not neglect pulse-width modulation (PWM) as a potential type of DAC, especially in microcontroller circuits where dedicated PWM peripherals are commonplace. Fig.6 shows the simplest possible configuration with a PWM output and an RC low-pass filter.

The time constant of the low-pass filter has to be much longer than the PWM period (T_{PWM}) to produce an average of the PWM waveform propor-

tional to its duty cycle. A longer time constant with respect to T_{PWM} means better averaging and a lower output ripple. The worst-case ripple occurs at 50% duty cycle and is given by $V_{RIPPLE} = V_{FS}(T_{PWM} \div 4RC)$.

The downside of having a long time constant (and lower ripple) is a slow response of the output voltage to changes in the PWM duty cycle.

Stephen Woodward published a really neat technique to address this problem in an EDN article published in 2017. Woodward showed that the time for the output voltage to settle to within the ripple voltage for a given change in duty cycle is $T_{SETTLE} = RC \times \log_e(V_{FS} \div V_{RIPPLE})$.

If we wanted to make a PWM DAC equivalent to a conventional DAC with 8-bit resolution, we would require that the peak-to-peak ripple be 1LSB (or $1/256$ of the full-scale voltage), corresponding to the quantising noise. The first equation above tells us this requires an RC time constant 64 times the PWM period.

The settling time (from the second equation) will therefore be $355 \times T_{PWM}$. Depending on what you are doing, that could be a long time! For 10kHz PWM, this is a settling time of 35ms. If you wanted a 10-bit resolution, the settling time would be even worse at 177ms.

Woodward's ingenious solution is shown at the bottom of Fig.6. Here, an inverted version of the PWM signal is injected into the output via a series RC network to cancel the ripple. I have shown this coming from a complementary PWM output on the microcontroller, but a logic gate inverter would work equally well.

If the new resistor and capacitor are the same value as the original ones, the ripple equation becomes $V_{\text{RIPPLE}} = \frac{1}{2}V_{\text{FS}} \times (T_{\text{PWM}} \div 4RC)^2$ and the settling time becomes $T_{\text{SETTLE}} = \frac{1}{2}T_{\text{PWM}} \times \sqrt{(V_{\text{FS}} \div V_{\text{RIPPLE}}) \times \log_e(V_{\text{FS}} \div V_{\text{RIPPLE}})}$.

Using the 8-bit example above, the RC time constant required to achieve the ripple target reduces from 64 to just four PWM periods, and the settling time reduces from $355T_{\text{PWM}}$ to $44T_{\text{PWM}}$ (or 4.4ms at 10kHz). This is an eightfold improvement in settling time!

In the 10-bit case, the settling time reduces to 11ms from 177ms for the original circuit. It turns out you don't even need precision components to achieve these improvements. It is sufficient to use 1% tolerance resistors and 10% tolerance capacitors. This is a circuit well worth knowing.

Other DAC types

If we agree that a duty-cycle-to-analog converter like the PWM example above is a form of DAC, we should also consider a frequency-to-voltage converter to be one too. Fig.7 shows a typical example using an LM331 IC. This circuit works as follows.

A square wave of frequency F_{IN} is differentiated by the RC network connected to pin 6, creating negative-going spikes on each falling edge. When these fall below the threshold set by the resistor divider connected to pin 7, the upper comparator sets the RS flip-flop.

When the flip-flop is set, the discharge transistor on pin 5 is switched off, and capacitor C_T begins to charge via R_T . When this voltage reaches $\frac{2}{3}V_{\text{CC}}$, the lower comparator resets the flip-flop, discharging C_T . This means the flip-flop is set for a fixed duration each time a falling edge on the input occurs.

The analog switch connected to the flip-flop's output directs a fixed current out of pin 1 during this period. The current is converted to a voltage by R_L and averaged by the $R_L C_L$ low-pass filter.

The level of this current is precisely controlled by the circuit connected to pin 2. An on-board bandgap reference and op amp ensure a precise 1.9V is always present on pin 2, meaning a current of $1.9V \div R_s$ flows out of this pin and therefore out of the current mirror to the analog switch.

A practical example

Fig.8 shows an extract of a circuit I designed some time ago for a precision instrument. This part of the circuit includes a voltage reference, a two-channel DAC and a couple of op amps. The output is intended to be a $\pm 2.0V$ precision voltage programmable by the microcontroller.

I will step through the operation and error analysis for this circuit, so it will be helpful to follow both the schematic and the error budget (Table 2). The operating temperature range for this device is 15-35°C, reflecting its intended use in a laboratory setting.

The MAX6225 provides a very stable, very accurate 2.5V reference ($\pm 200\text{ppm}$ initial accuracy, $\pm 2\text{ppm}/^\circ\text{C}$). An inverted copy of the reference is provided by the inverting amplifier (IC1), an LTC2057 zero-drift op amp. The inverting op amp uses relatively high value resistors (10k Ω) to minimise the load on the precision reference.

These resistors are high-precision (0.01%) types with very low thermal drift ($\pm 5\text{ppm}/^\circ\text{C}$), since we don't want to compromise the performance of an expensive reference. The LTC2057 has very low input offset voltage ($\pm 4\mu\text{V}$) and even lower offset drift ($\pm 15\text{nV}/^\circ\text{C}$), as we would expect from a "zero drift" op amp.

Because we have used relatively high value resistors, I have included the op amp's input offset current error in the table on line 3. You can see from line 4 of the table that the total error at the input of the reference inverter is dominated by the reference error and is 0.02%.

You can see why I selected 0.01% resistors for this circuit – to keep the gain error down to a similar order of magnitude as the reference error. This leads to a total error for the negative reference of 0.047% over the temperature range, compared to a total error of 0.022% for the positive reference.

I was not 100% happy with the negative reference error, but figured that as I was only making a handful of these

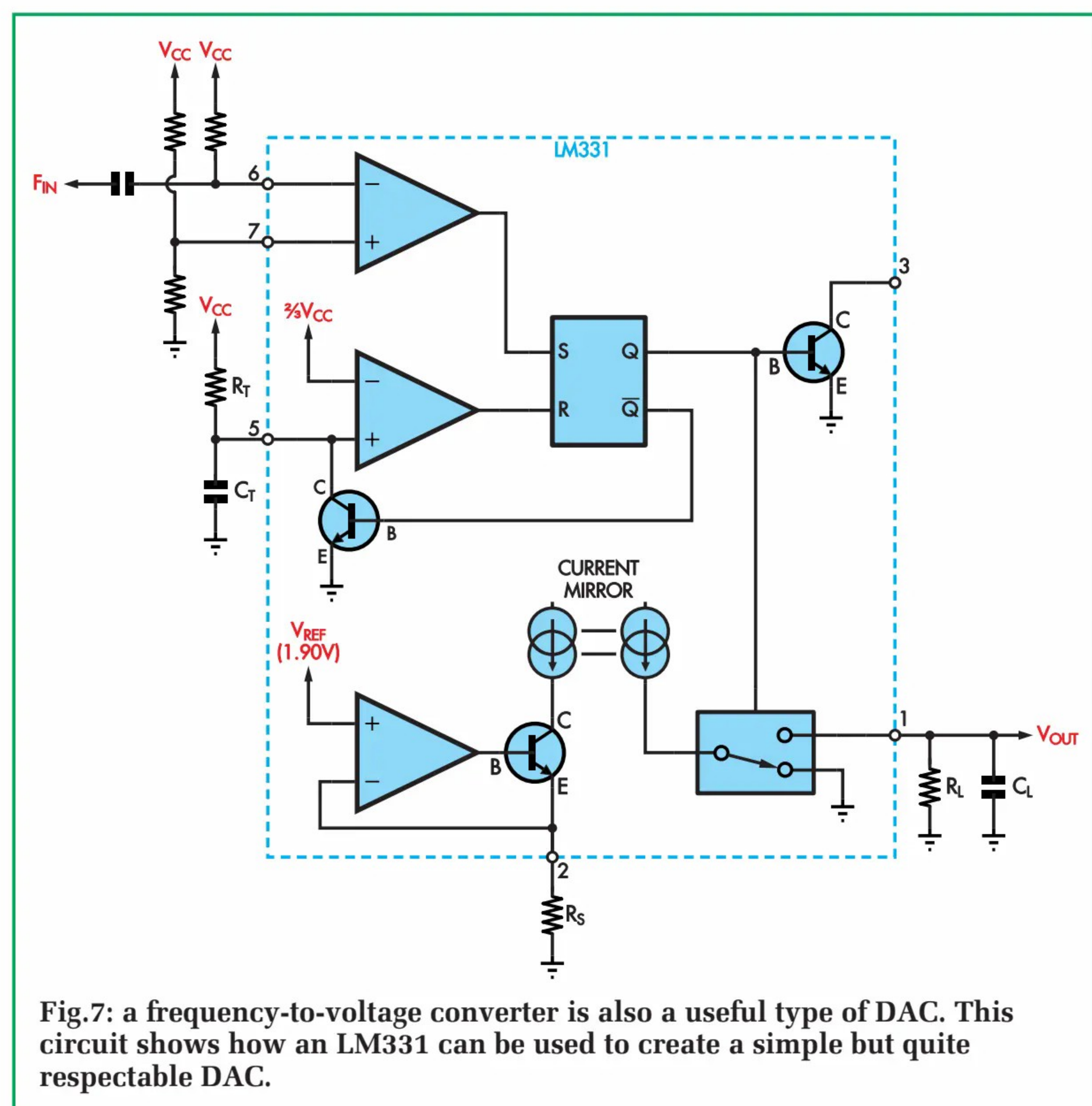


Fig.7: a frequency-to-voltage converter is also a useful type of DAC. This circuit shows how an LM331 can be used to create a simple but quite respectable DAC.

Digital-to-Analog Conversion (DACs)

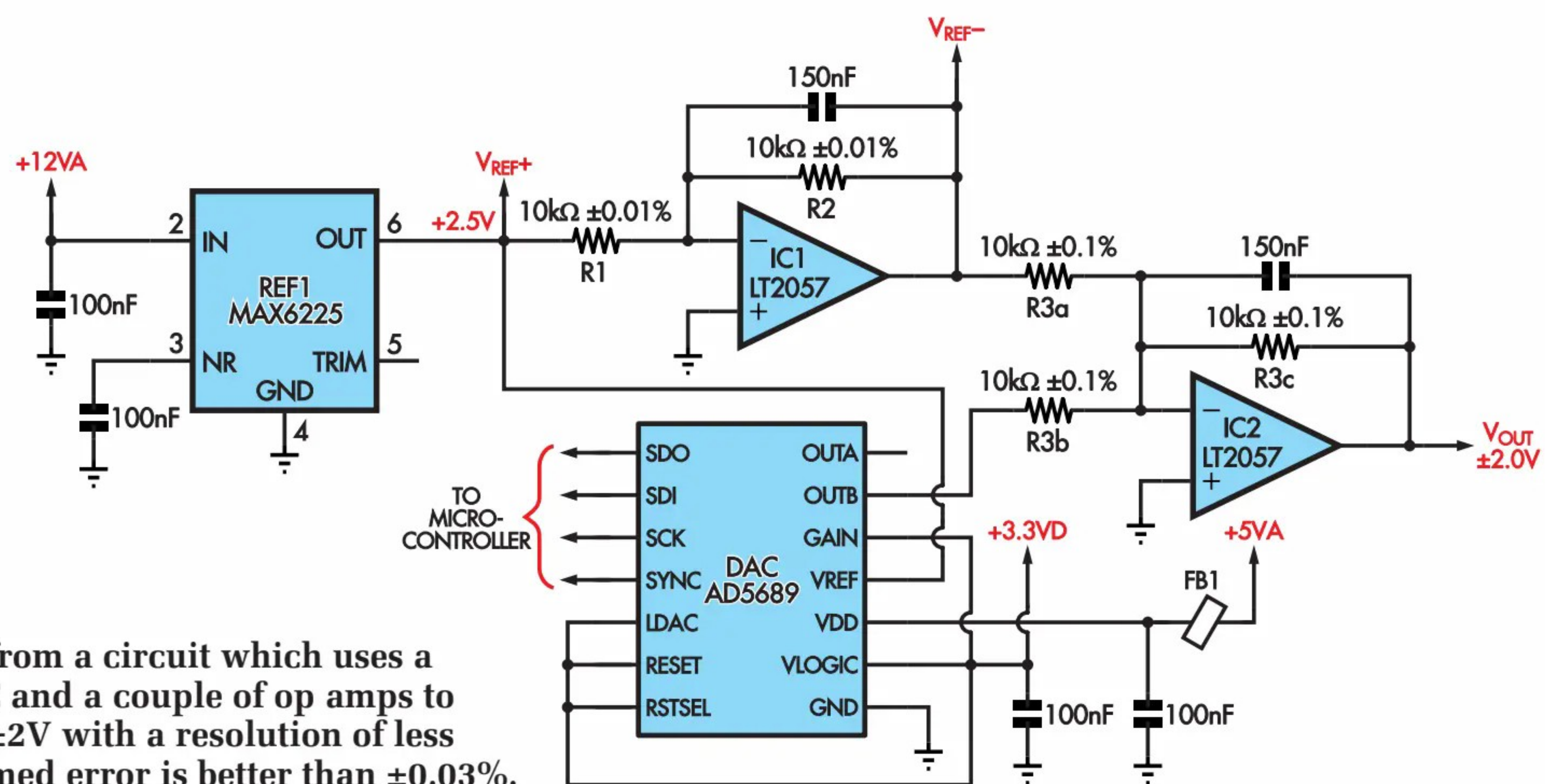


Fig.8: this is an excerpt from a circuit which uses a voltage reference, a DAC and a couple of op amps to create a programmable $\pm 2\text{V}$ with a resolution of less than $200\mu\text{V}$. The untrimmed error is better than $\pm 0.03\%$.

devices, my results would likely be much better. The odds of two identical resistors being at the opposite extremes of tolerance are low enough that if I did find an outlier, I could manually select resistors to fix it.

The measured results show both references to be within 0.01% of each other on the prototype.

The positive 2.5V reference is applied to the DAC. I used a very nice dual-channel, 16-bit resistor-string DAC. It can be configured for a gain of one or two. I used a gain of two to make the subsequent circuit design simpler. This means the full-scale output is 5.0V.

In this configuration, the DAC has

an offset error of $\pm 1.5\text{mV}$, a gain error of 0.1% and an INL of $\pm 1\text{LSB}$. Thanks to the manufacturer for specifying the three key figures in three different ways!

That said, this is pretty good performance for a DAC, especially the INL. I converted all these figures to relative errors in the table and added them using the root-sum-of-squares method to arrive at a TUE of 0.104%, dominated by the gain error. The data sheet actually provides a TUE figure of 0.1%, so I did this exercise to just demonstrate how TUE is calculated. It is nice when theory and reality agree!

Since the voltage at the output of the DAC is the reference multiplied through the DAC coefficient, the total error is calculated as the sum of relative errors on line 12. We get a total error here of 0.127% over the full temperature range ($0.124\% + 0.003\%$), dominated by the DAC gain error of 0.1%.

I'm not usually a proponent of 'typical' specifications, but if our DAC was within the typical range, the total error would be closer to 0.05%.

Finally, the 0-5V DAC output is summed with the negative reference (and inverted) by IC2. This produces an output voltage that ranges from +2.5V when the DAC code is zero (0x0000),

Table 2: positive/negative voltage references, DAC & offset amp errors

Table 2: positive/negative voltage references, DAC & offset amp errors			At Nominal 25°C		Additional error over 15-35°C (Nominal ±10°C)	
	Error	Nominal Value	Abs. Error	Rel. Error	Abs. Error	Rel. Error
1	MAX6225ACASA+ (±200ppm, ±2ppm/°C)	2.5V	500µV	0.02%	50µV	0.002%
2	Op Amp: LTC2057 (V _{os} ±4µV, 15nV/°C)	0V	4µV		100nV	
3	Op Amp I _{os} × 10kΩ 10kΩ: LM7301 (I _{os} ±400pA, ±1pA/°C)	0V	2µV		50nV	
4	Voltage at Op Amp Input (Line 1 + Line 2 + Line 3)	2.5V	506µV	0.02%	50.2µV	0.002%
5	Op Amp Gain: R/R Stackpole RNCF0603TKY10K00 (0.01%, 5ppm/°C)	1		0.02%		0.005%
6	+V_{REF} error (Line 1)	2.5V	500µV	0.02%	50µV	0.002%
7	-V_{REF} error (Line 4 × Line 5)	-2.5V	1mV	0.04%	175.2µV	0.007%
8	DAC Offset error: AD5689 (±1.5mV, ±1µV/°C)	5V	1.5mV	0.03%	10µV	0.000%
9	DAC Gain error: AD5689 (0.1%, ±1ppm/°C)	2		0.1%		0.001%
10	DAC Linearity: AD5689 (INL ±1LSB, DNL ±1LSB)	0		0.002%		0.000%
11	DAC total unadjusted error (root sum of squares Lines 8-10)	5V	5.2mV	0.104%	51µV	0.001%
12	DAC total error (Line 6 × Line 11)	5V	6.2mV	0.124%	620µV	0.003%
13	Op Amp: LTC2057 (V _{os} ±4µv, 15nV/°C)	0V	4µV		150nV	
14	Op Amp I _{os} × 1kΩ 1kΩ 1kΩ: LM7301 (I _{os} ±400pA, ±1pA/°C)	0V	133.2nV		3.3nV	
15	Op Amp Gain: R/R Stackpole ACASA1002U1002P1AT (0.05%, 5ppm/°C)	1		0.05%		0.005%
16	Voltage at Op Amp Input (Line 7 + Line 12 + Line 13 + Line 14)	2.5V	7.2mV	0.29%	250.6µV	0.01%
17	Voltage at Op Amp Output (Line 15 × Line 16)	2.5V	8.5mV	0.34%	375.6µV	0.015%



The Wireless for the Warrior books are references for the history and development of radio communication equipment used by the British Army from the very early days of wireless up to the 1960s.

Volumes 1 & 3 are still available. Order a printed copy now from:
<https://pemag.au/link/ac20>

to almost -2.5V when the DAC code is full scale (0xFFFF).

The error budget for this circuit is similar to the previous examples. This time, I used low-cost resistor arrays with $\pm 0.05\%$ tolerances on the matching. If I had used individual 0.01% resistors for this, there would be a 0.03% error due to there being three resistors involved. The improvement of 0.02% does not justify the extra cost given the other sources of error in the circuit.

There are a couple of things worth noting before we discuss the results this circuit produced. First, you will see that I am only using a range of $\pm 2V$, not the full $\pm 2.5V$ the circuit is capable of. This is because near zero and full scale, the DAC output is prone to errors, since these are right at its power supply rails. We already know that no output can truly swing all the way to the rails.

Avoiding the ends of the span costs us some precision, since the whole range of codes is not used. In this case, the valid codes are 6553 to 58,981 for a 4V span, giving us a resolution of about $191\mu V$, which is plenty for my application.

You should always avoid the extremes of DAC & ADC ranges in precision applications. You can go much closer than I have here, but there will probably be errors right at the edges.

I have also taken a lot of care with the power supplies. It is not worth spending good money on precision components and skimping on the power supply. The digital and analog supplies come from separate linear regulators. I included a ferrite bead on the analog supply to the DAC, more to protect the rest of the circuit from glitches caused by the DAC switching than vice versa.

Results

I first measured the positive and negative references and found they were $+2.50015V$ and $-2.49989V$, both well under 0.01% away from the nominal value and within 0.01% of each other.

I measured the output voltages at code intervals of 400 hexadecimal (1024 decimal) over the full range. With a code of zero, I measured $+2.49930V$ (0.028% error). At $0x8000$, I measured $481\mu V$ (0.019% error), while at full

scale ($0xFFFF$), I measured $-2.45720V$ (-1.7% error). As mentioned above, we expect the extremes to be poor.

If we look at the range of interest, the error is never worse than $+0.024\%$; in fact, it is also never less than $+0.018\%$, suggesting we have an offset error, albeit a small one. Sure enough, the absolute error ranges from $460\mu V$ to $610\mu V$ and averages $550\mu V$.

Can we trim this error somehow? My circuit also includes an ADC, and switching that allows me to measure the voltage we are concerned with. If we were to measure the voltage with the code $0x8000$ (corresponding with $0V$ out), we would be able to measure the $481\mu V$ offset and correct for it in software.

We could similarly measure the voltage at either end of the span ($\pm 2V$ nominal) and correct for that. This is easier said than done, and we will look into it in more detail in a later article.

I also performed a full noise analysis of this circuit, which I have included in Table 3. I won't go into the gory details since I used the same techniques I described in the last article. Overall, the RMS noise voltage should be around $1.4\mu V$ over a $10Hz$ bandwidth. That does not include quantisation noise, since this is a DC application.

The biggest contributor is the DAC noise, with the reference coming in next. Given the $191\mu V$ resolution of the DAC, this level of noise is not going to impact the precision of our circuit.

In summary, we can almost certainly get to an overall precision close to that of the reference at 0.02% , and it is pretty hard to do any better than that!

PE

Table 3 – noise analysis with $10Hz$ nominal bandwidth

Noise Source	Notes	Noise Voltage (RMS)
1 Positive reference noise MAX6225 ($15nV/\sqrt{Hz}$, $f_c 100Hz$)	f_b straddles f_c so use $f = f_b + f_c \log_e(10) = 240Hz$	232.4nV
2 Ref inverter amp voltage noise: LTC2057 ($11nV/\sqrt{Hz}$)	Data shows noise flat from $0.1Hz$ to $10Hz$	34.8nV
3 Ref inverter amp voltage noise: LTC2057 ($170fA/\sqrt{Hz}$)	$10k\Omega \parallel 10k\Omega$ resistors in inverting input	2.68nV
4 Ref inverter $10k\Omega$ input resistor & feedback resistor	$\sqrt{4kTRf_b}$	40.7nV
5 Negative reference noise	Line 1 + Noise Gain (2) $\times$ Line 2 #	237.5nV
6 DAC noise AD5689 ($300nV/\sqrt{Hz}$)	Use $f_b = 10Hz$	948.7nV
7 DAC output noise	Line 6 + Line 1	1.2 μV
8 Summing amp voltage noise: LTC2057 ($11nV/\sqrt{Hz}$)	Data shows noise flat from $0.1Hz$ to $10Hz$	34.8nV
9 Summing amp voltage noise: LTC2057 ($170fA/\sqrt{Hz}$)	$10k\Omega \parallel 10k\Omega \parallel 10k\Omega$ resistors in inverting input	1.79nV
10 $10k\Omega$ input resistors & feedback resistor	$\sqrt{4kTRf_b}$	40.7nV
11 Scaled output noise	Line 7 + Line 5 #	1.4 μV

other errors are at least an order of magnitude smaller, so they can be ignored

Practical Electronics PCB SERVICE

PROJECT	CODE	PRICE
APRIL 2024		
Basic RF Signal Generator.....	CSE221001	£8.99
Basic RF Signal Generator front panel (black solder mask) ..	CSE220902B	£9.99
Silicon Chirp Cricket.....	08101231	£7.99
Test Bench 'Swiss Army Knife'	04110221	£9.99
MARCH 2024		
Digital Volume Control Potentiometer SMD version	01101231	£8.95
Digital Volume Control Potentiometer through-hole version .	01101232	£9.95
Advanced SMD Test Tweezers body.....	04106221	£6.95
Advanced SMD Test Tweezers legs (pair)	04106212	£7.95
FEBRUARY 2024		
Active Mains Soft Starter	10110221	£10.95
JANUARY 2024		
Q Meter	CSE220701	£8.95
Q Meter front panel (black solder mask)	CSE220704	£7.95
Raspberry Pi Pico W BackPack	07101221	£8.95
DECEMBER 2023		
Digital Boost Regulator	24110224	£8.95
Dual-Channel Power Supply for Breadboards.....	04112221	£11.95
Display Adaptor for the Breadboard PSU	04112222	£9.95
NOVEMBER 2023		
LC Meter Mk3.....	CSE220503C	£9.95
DC Supply Filter for vehicles.....	08108221	£8.95
Discrete Audio Op Amp PCB.....	AO1-JUL23	£9.95
Discrete Audio Op Amp PCB + essential components.....	AO2-JUL23	£17.95
OCTOBER 2023		
Buck/Boost Charger Adaptor	14108221	£9.95
PIC Breakout Board for SOIC parts.....	24110225	£5.95
PIC Breakout Board for DIP parts	24110222	£5.95
AVR64DD32 Breakout board.....	24110223	£5.95
Automatic Train Controller.....	09109221	£6.95
Chuff Sound module	09109222	£6.95
SEPTEMBER 2023		
Mini LED Driver	16106221	£7.95
New GPS-Synchronised Clock.....	19109221	£9.95
AUGUST 2023		
110dB RF Attenuator	CSE211003	£9.95
Universal Battery Charge Controller (2023 update)	14107192	£9.95
Wide-Range OhmMeter	04109221	£12.95
JULY 2023		
Multimeter Checker/Calibrator	04107221	£10.95
JUNE 2023		
Arduino Programmable Load.....	04105221	£8.95
Buck-Boost LED Driver	16103221	£8.95
MAY 2023		
Precision AM-FM DDS Signal Generator	CSE211002	£14.95
Improved SMD Test Tweezers programmed PIC.....	0410621PIC	£12.95
AO 2x-dual-to-quad through-hole (pack of five).....	AO1-MAY23	£6.95
AO 2x-dual-to-quad SMD (pack of five).....	AO2-MAY23	£6.95
AO 2x-single-to-dual through-hole (pack of five).....	AO3-MAY23	£6.95
AO 2x-single-to-dual SMD (pack of five).....	AO4-MAY23	£6.95

PROJECT	CODE	PRICE
APRIL 2023		
500W Amplifier Module.....	see p.22, April 2023	
Clipping Indicator (per channel)	01112211	£7.95
CD Welder Power Supply (one needed).....	29103221	£9.95
CD Welder Controller (one needed).....	29103222	£9.95
CD Welder Energy Storage module (several needed).....	29103223	£7.95
AO Universal Dual Op Amp Board	AO1-APR23	£9.95
AO Stereo RIAA precision passives kit for dual op amp.....	AO2-APR23	£8.95
MARCH 2023		
Pico BackPack	07101221	£9.95
Semaphore Signal (controller).....	09103221	£7.95
Semaphore Signal (blade).....	09103222	£5.95
FEBRUARY 2023		
Tesla Coil driver board.....	26102221	£9.95
Tesla Coil potentiometer board.....	26102222	£5.95
Cooling Fan Controller & Loudspeaker Protector.....	01102221	£8.95
Remote Gate Controller	11009121	£12.95
JANUARY 2023		
Classic LED Metronome – 8-LED.....	23111211	£7.95
Classic LED Metronome – 10-LED.....	23111212	£8.95
Multi-Channel Speaker Protector – 6 channel	01101221	£9.95
Multi-Channel Speaker Protector – 4 channel	01101222	£7.95
Remote Control Range Extender – IR-to-UHF	15109212	£5.95
Remote Control Range Extender – UHF-to-IR	15109211	£7.95
AO Universal Single Op Amp Board.....	AO1-JAN23	£7.95
DECEMBER 2022		
Hummingbird Amplifier	01111211	£9.95
SMD Trainer PCB	29106211A	£8.95
SMD Trainer PCB + parts.....	29106211B	£13.95
NOVEMBER 2022		
USB Cable tester – main PCB	04108211	£12.95
USB Cable tester – front panel	04108212	£5.95
USB Cable tester – optional panel	SC5970	£5.95
Model Railway Carriage Lights – PCB	09109211	£6.95
AO trnsfmr PCB – standard VTX-A range	VTX-101-007	£6.95
AO trnsfmr PCB – dual-outline VTX102-3001/101-3002	VTX-Dual	£6.95
OCTOBER 2022		
SMD Test Tweezers – PCB and pair of tweezer arms	04106211-2	£11.95
SMD Test Tweezers – programmed PIC12F1572-I/SN	0410621A	£7.95
Tele-com.....	12110211	£12.95
SEPTEMBER 2022		
Touchscreen Digital Preamp – main board	01103191	£12.95
Touchscreen Digital Preamp – adaptor board pair	01103192	£5.95
20A DC Motor Speed Controller.....	11006211	£9.95
AUGUST 2022		
Multi-purpose Battery Manager – I/O Expander.....	11104212	£5.95
Multi-purpose Battery Manager – Switch Module.....	11104211	£8.95
Simple MIDI Music Keyboard (for 8 switches)	23101213	£6.95
Nano Pong.....	08105212	£7.95
JULY 2022		
Silicon Labs AM/FM/SW Radio	CSE210301C	£10.95
Level Crossing Controller	09108211	£6.95
JUNE 2022		
Full-wave Motor Speed Controller.....	1010221	£8.95
PIC Programming Helper for 8-pin PICs only	24106211	£7.95
PIC Programming Helper for 8, 14 or 20-pin PICs	24106212	£10.95
Advanced GPS Computer	05102211	£9.95

Silicon Chip kits & parts for projects in this issue (posted from Australia):

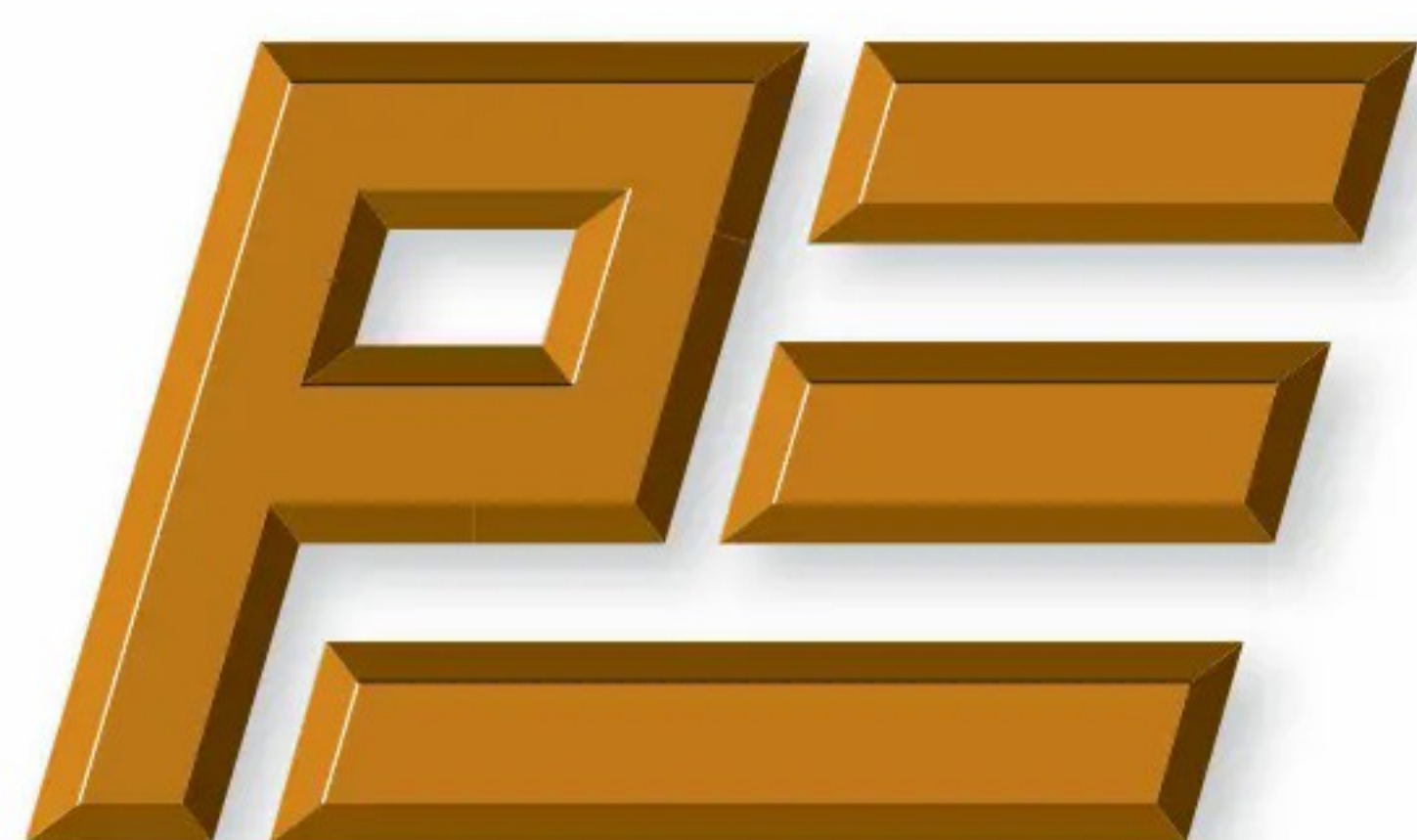
Dual Mini LED Dice kit (TH LED version)
Dual Mini LED Dice kit (SMD LED version)
Hard-to-get parts for Dual Hybrid Power Supply regulator board
Hard-to-get parts for Dual Hybrid Power Supply CPU board

<https://siliconchip.au/Shop/20/6849>
<https://siliconchip.au/Shop/20/6961>
<https://siliconchip.au/Shop/20/6096>
<https://siliconchip.au/Shop/20/6121>

AUD \$21 (about £10.50) + P&P
AUD \$21 (about £10.50) + P&P
AUD \$120 (about £60) + P&P
AUD \$72 (about £36) + P&P

20% VAT is included in the above prices (it will be removed for customers outside the UK). Postage & packing cost is typically around £12-15.
Please allow 1-2 weeks for delivery (longer in some cases).

CLASSIFIED ADVERTISING



Practical Electronics

If you want your advertisements to be seen by the largest readership at the most economical price then our classified page offers excellent value.

The rate for semi-display space is £10 per centimetre high, with a minimum height of 2.5cm. All semi-display advertisements have a width of 5.5cm. The prepaid rate for classified adverts is 40p per word (minimum 12 words).

For rates, further information on display and classified advertising or to book an ad, please email us at pe@pemag.au

Practical Electronics reaches more UK readers than any other UK monthly hobby electronics magazine. Our sales figures prove it.

We have been the leading monthly magazine in this market for more than thirty years.

BOWOOD ELECTRONICS LTD

Suppliers of Electronic Components

www.bowood-electronics.co.uk

Unit 10, Boythorpe Business Park, Dock Walk, Chesterfield, Derbyshire S40 2QR. Sales: 01246 200 222

Send large letter stamp for Catalogue

Electrical Industries Charity (EIC)

We help people working in the electrical, electronics and energy community as well as their family members and retirees.

We use workplace programmes that give the industry access to financial grants and a comprehensive range of free and confidential services.

www.electricalcharity.org



COAST ELECTRONICS

BREAKOUTS-COMPONENTS-
CONTRACT DESIGN-3D PRINTER PARTS-
MUSICAL-MICROCONTROLLERS

WWW.COASTELECTRONICS.CO.UK

MISCELLANEOUS

VALVES AND ALLIED COMPONENTS?

For free stock list and/or advice, please contact me: geoffdavies337@gmail.com Telephone: 01788 574774

PIC DEVELOPMENT KITS, DTMF

kits and modules, CTCSS Encoder and Decoder/Display kits.

Visit www.cstech.co.uk

COMPONENTS AS JOB LOT

Large stock. Transistors, Diodes, Logic, Memory and Analogue parts. 1000s of passive items. Catalogued and sorted. 'New Old Stock' in racks. Email: rojobarn@gmail.com

Andrew Kenny – Qualified Patent Agent

EPO UKIPO USPTO

Circuits Electric Machinery Mechatronics

Web: www.akennypatentm.com

Email: Enquiries@akennypatentm.com

Tel: 0789 606 9725

ADVERTISING INDEX

AOSHOP	49
BACK ISSUE PDFs	41
BASIC MICRO	19
ENIGMA COMPONENTS SHOP	40
HAMMOND ELECTRONICS Ltd.	13
MICROCHIP TECHNOLOGY	Outer back cover
PCBWAY	Inner front cover
POLABS	63
TAG-CONNECT	40
TEACH-IN 9	55
TERRINGTON COMPONENTS	19
WIRELESS FOR THE WARRIOR BOOKS	63, 76

Advertising mailing address

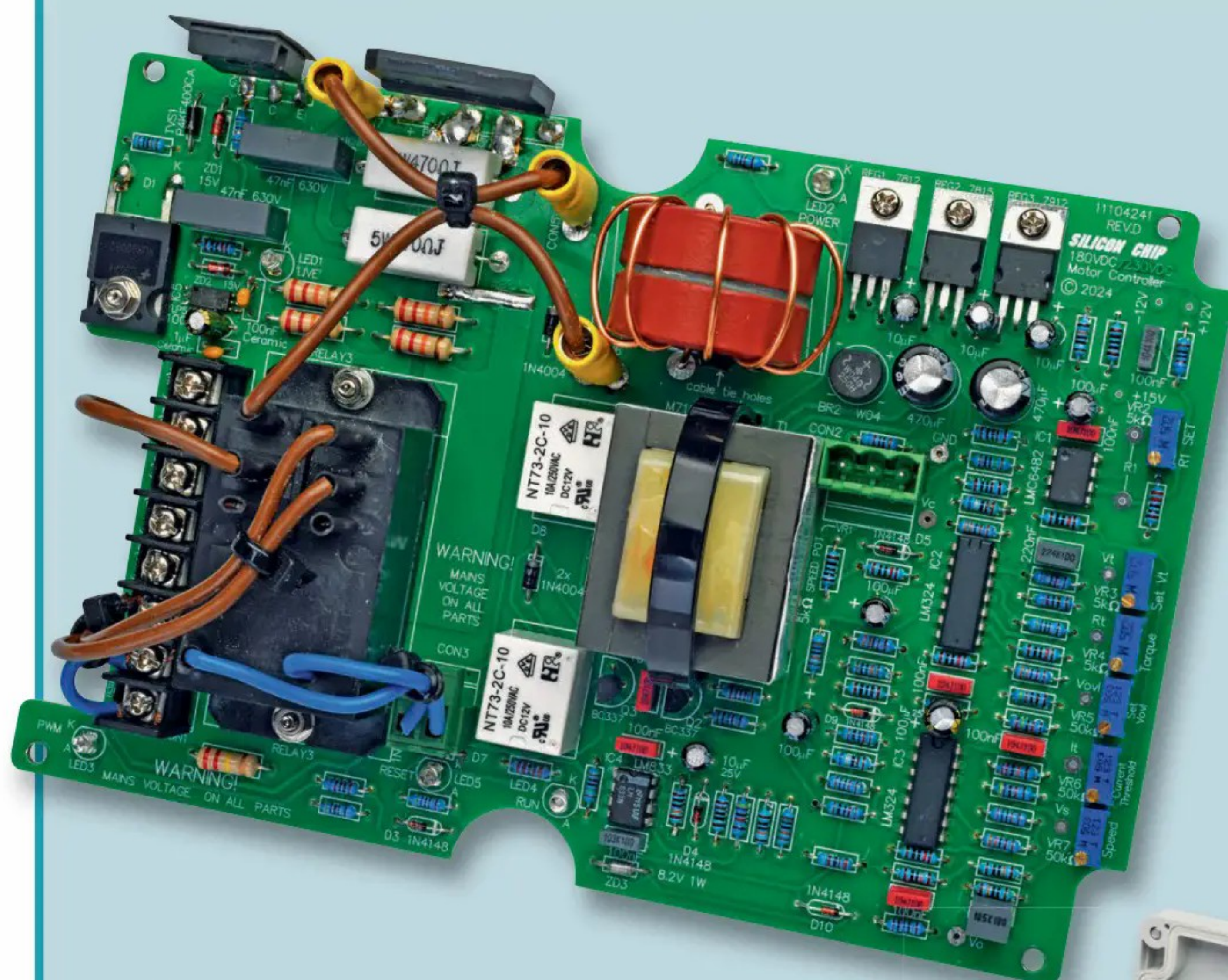
Electron Publishing Australia
PO Box 194
Matraville NSW 2036
Australia

Email pe@pemag.au

Web www.electronpublishing.com

For editorial contact details, see page 2.

Next Month – in the July issue



180-230V DC Motor Speed Controller

Controls the speed of treadmill and lathe motors, among others.

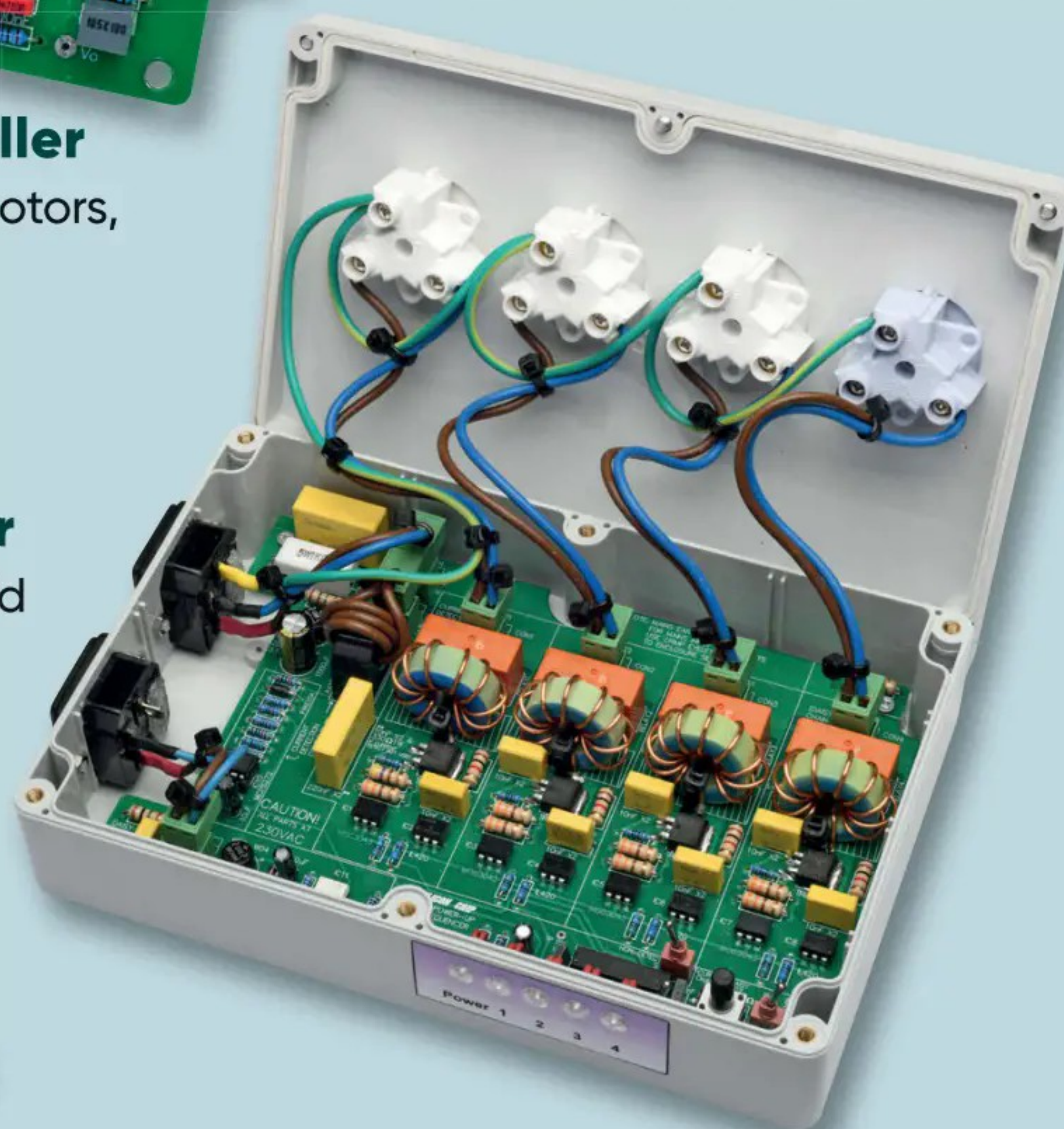
Another Use for the Mains Power-Up Sequencer

Sequential load power-up to avoid generator overload.



Compact OLED Clock & Timer

A convenient way to provide timed alerts without draining your smartphone battery.



On sale 5th of June, 2025

Content may be subject to change

Practical Electronics

Subscriptions hotline

To subscribe or renew online, paying by credit/debit card or PayPal, please visit our shop at:
www.electronpublishing.com

For those wishing to pay by direct debit, cheque or over the telephone, please call 'Select Publisher Services' on 01202 087631, or send a cheque (payable to: 'Select Publisher Services Ltd') with your details to:

Practical Electronics Subscriptions, PO Box 6337, Bournemouth BH1 9EH, United Kingdom

Remember, we print the date of the last issue of your current subscription in a box on the address sheet that comes with your copy.

Digital subscribers, please call 07918 614662 or visit: www.electronpublishing.com

Published on approximately the first Thursday of each month by Silicon Chip Publications, U1 234 Harbord Road Brookvale NSW 2100 Australia. Printed in England by The Manson Group, Reynolds House, 8 Porters Wood, Valley Road Industrial Estate, St Albans AL3 6PZ. Distributed by Seymour, 86 Newman St, London W1T 3EX. Subscriptions UK: £33.99 (6mo); £59.99 (12mo.); £114.99 (2yrs). EUROPE: airmail service, £38.99 (6mo); £69.99 (12mo); £129.99 (2yrs). REST OF THE WORLD: airmail service, £46.99 (6mo); £84.99 (12mo); £164.99 (2yrs). Make payments to 'Practical Electronics', Practical Electronics Subscriptions, PO Box 6337, Bournemouth BH1 9EH United Kingdom. Email: pesubs@selectps.com. PRACTICAL ELECTRONICS is sold subject to the conditions that it shall not, without the written consent of the Publishers first having been given, be lent, resold, hired out or otherwise disposed of by way of Trade at more than the recommended selling price on the cover, and that it shall not be lent, resold, hired out or otherwise disposed of in a mutilated condition or in any unauthorised cover by way of Trade or affixed to or as part of any publication or advertising, literary or pictorial matter.

Did you know our online shop now sells the current issue of *PE* for £5.99 inc. p&p?

Practical Electronics

The UK's premier electronics, computing and maker magazine

Circuit Surgery
Implementing DSP
with a microcontroller

Audio Out
Cascoding JFETs and BJTs
in a direct injection box

Precision Electronics
Bringing analog and digital
together: ADCs & DACs

**Dual Hybrid
(switch-mode + linear)
Bench Supply**

The Styloclone
A simple electronic
musical instrument

**Two kinds of
Mini LED Dice**

PLUS!

Techno Talk – Humanoid robots may be a reality soon
Cool Beans – shift registers and LED bargraphs
The History of Electronics – our final instalment

Jun 2025 £5.99



www.electronpublishing.com |  @practicalelec |  practicalelectronics

Practical Electronics

The UK's premier electronics, computing and maker magazine

Precision Electronics
Quantifying and
dealing with noise

**DIY USB-C
serial
adaptor**

Il Tester 9000
ke on an old game

May 2025 £5.99



You read that right! We now sell the current issue of your favourite electronics magazine for exactly the same price as in the High Street, but we deliver it straight to your door – and for UK addresses we pay the postage. No need to journey into town to queue outside the newsagent. Just go to our website, set up an account in 30 seconds, order your magazine and we'll do the rest.

www.electronpublishing.com



Low-Cost High Safety Microcontroller: AVR[®] SD

ASIL C and SIL 2 Compliant

Achieving high safety integrity levels (SIL) for safety-critical functions in automotive and industrial applications is complex and requires expertise and rigorous processes. Functional safety engineers face pressure to reduce design costs to stay competitive. Selecting the right microcontroller is crucial to meet international functional safety standards and optimize costs without compromising safety. To address this, we announced the AVR[®] SD family, compliant with ISO 26262 (ASIL C) and IEC 61508 (SIL 2). It is ideal for safety functions like monitoring temperature, current, or voltage spikes and simplifies designs by offloading safety-critical functions from a central processor. With built-in hardware safety features (dual cores in lockstep, ECC on memory, a dedicated error controller), it detects faults within 1 millisecond and includes a complete development ecosystem with compliant software, safety documentation, and a certified compiler.

Key Features

- **Dual-Core Lockstep CPU:** Ensures high diagnostic coverage of faults by running two identical processing cores in parallel and catching any discrepancies almost instantaneously (1 clock cycle).
- **Error Correcting Code (ECC) on All Memories:** Detects and corrects errors, maintaining data integrity.
- **Dedicated Error Controller Module:** Continuously monitors for errors and takes corrective actions, maintaining system integrity and safety.
- **Voltage and Clock Monitors:** Ensures the microcontroller operates within safe voltage and clock parameters, preventing system failures due to power or timing issues.
- **Fault Detection Time Intervals:** Suitable for applications with high safety demands and fault detection time interval targets down to 1 millisecond.

Join the Future of Safety-Critical Design With AVR32SD32.



microchip.com/avr-sd



The Microchip name and logo and the Microchip logo are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries. All other trademarks are the property of their registered owners.
© 2025 Microchip Technology Inc. All rights reserved.
MEC2610A-UK-04-25